FizzBuzz EDSAC **Olve Maudal**



A 5 minute lightning talk at ACCU 2015, April 24, Bristol, UK



https://youtu.be/x-vS0WcJyNM

The EDSAC 1951 film abridged version Commentary by M. V. Wilkes

EDSAC Initial Orders and Squares Program

Martin Richards



The Squares Program

The following is an annotated listing of the program

This program, written by Maurice Wilkes in June 1949, outputs the following table of sq numbers 1 to 100



EDSAC

EDSAC (Electronic Delay Storage Automatic Computer), pictured below, was the world's first stored-program computer to operate a regular computing service. Maurice Wilkes lead the team responsible for its design and construction. It ran its first program successfully on May 6, 1949.



EDSAC's main memory used mercury delay lines to hold 512 words of 35 bits. We will use the notation: w[0]. EDSACS main memory used mercury delay mes to noid 512 words of 55 bits. We will use the notation: $w[0]_1$, $w[0]_2$, $w[0]_2$ to refer to these words of memory. Each word could be split into two 1-5-bit halves, separated by a padding bit. We will use the notation m[a], a = 0, 1, ..., 1023 to represent these 17-bit memory locations. The word at address 2n, namely $w[2n]_2$, consisted of the concatenation of m[2n + 1], a padding bit, and m[2n]. Note that m[1] is the senior half of w[0].

$$w[2n]: \begin{array}{c|c} 1/ & 1 \\ \hline & 1/ \\ \hline \hline & 1/ \\ \hline & 1/ \\ \hline & 1/ \\ \hline & 1/ \\ \hline &$$

The machine had two central registers visible to the user: the 71-bit accumulator and the 35-bit multiplier register. We will use the notation ABC to represent the whole accumulator, and A and AB to represent its senior 17 and 35 bits, respectively. We will use RS to represent the whole multiplier register and R to represent its senior 17 bits. The leftmost bit of each register was the sign bit and the remaining bits form a binary fraction.

EDSAC's machine instructions (also called orders) occupied 17 bits. The leftmost 5 bits was the operation code, the next

Drder format:
$$\begin{array}{c|c} 5 & 1 & 10 & 1\\ \hline \ast \ast \ast \ast \ast \end{array}$$

$$\begin{array}{c|c} \hline & & \\ Op & Unused & Address & S/L \end{array}$$

Orders were punched on paper tape and consisted of: a character that directly gave the 5-bit operation code, followed by zero or more decimal digits giving the address, and terminated by S or L specifying the operand length bit. For example, R16S assembled to 00100 0 0000010000 0 and T11L to 00101 0 0000001011 1 . Note that the characters R and T had codes 4 and 5, respectively

The Character Set

EDSAC used 5-bit integers (0 to 31) to represent characters using two shifts: letters and figures. In letter shift the codes 0

The 1949 Instruction set

EDSAC's instructions in 1949 was very simple and were executed at a rate of about 600 per second. They were as follows:

AnS:	A += m[n]	AnL:	AB += w[n]
SnS:	A = m[n]	SnL:	AB = w[n]
HnS:	R += m[n]	HnL:	RS += w[n]
VnS:	AB += m[n] * R	VnL:	ABC $+= w[n] * RS$
NnS:	AB -= m[n] * R	NnL:	ABC $-= w[n] * RS$
TnS:	m[n] = A; ABC = 0	TnL:	w[n] = AB; ABC = 0
UnS:	m[n] = A	UnL:	w[n] = AB
CnS:	AB += m[n] & R	CnL:	ABC += w[n] & RS
RnS, RnL:	Shift ABC right arithmetically by the r	number	of places corresponding to the
	position of the least significant one i	in the s	shift instruction. For example,
	ROL, R1S, R16S and ROS shift by 1, 2	, 6 and	15 places, respectively.
LnS, LnL:	Shift ABC left arithmetically by the n	umber	of places corresponding to the
	position of the least significant one i	n the s	shift instruction. For example,
	LOL, L1S, L16S, L64S and LOS shift	: by 1,	2, 6, 8 and 13 places, respec-
	tively.		
EnS:	if $A \ge 0$ goto n		
GnS:	if $A < 0$ go o n		
InS:	Place the next paper tape character in	the lea	ast significant 5 bits of $m[n]$.
0 <i>n</i> S:	Output the character in the most sign	ificant	5 bits of m[n].
FnS:	Verify the last character output.		

- XnS:No operation YnS:
- Add a one to bit position 35 of ABC, counting the sign bit as bit zero. This effectively rounds ABC up to 34 fractional bits. Stop the machine and ring a bell.
- ZnS

The numerical values in the accumulator and multiplier registers are normally thought of as signed binary fractions, but integer operations could also be done easily. For example, the order VIS can be interpreted as adding the product of the 17-bit signed integer in $\mathbb{R}[1]$ and to the 17-bit integer in $\mathbb{R}[1]$ and to the 17-bit integer in RS and adding the result into bits 0 to 32 of the ABC. With a suitable shift, the integer result can be placed in the senior 17 bits of A ready for storing in memory.

Initial Orders

The four glass panels on your right contain 20 segments of 5 track paper tape. Reading from right to left and from top to bottom, the first five segments correspond to the initial orders, and the remaining 15 to a program to compute squares. The glass panels contain errors so a corrected version of the panels are given below.

The initial orders were written by David Wheeler in May 1949 to load and enter a paper tape represention of a program. When EDSAC was started, these initial orders were placed in memory locations 0 to 30 by a mechanism involving uniselect tors before execution stared from location 0.

The glass panels give a paper tape representation of these orders even though no such paper tape ever existed. The following is an annotated listing of this program.

Order bit pattern	Loc	Order	Meaning	Comment
00101 0 000000000 0	0:	TOS	m[0]=A; ABC=0	
10101 0 000000010 0	1:	H2S	R=m[2]	Put 10<<11 in R
00101 0 000000000 0	2:	TOS	m[0]=A; ABC=0	
00011 0 0000000110 0	3:	E6S	goto 6	Jump to main loop
00000 0 000000001 0	4:	P1S	data 2	The constant 2
00000 0 000000101 0	5:	P5S	data 10	The constant 10
00101 0 000000000 0	6.	TOS	m[0]=A · ABC=0	Start of the main loon
01000 0 000000000 0	7:	TOS	m[0] = rdch()	Get operation code
11100 0 000000000 0	8:	AOS	A+=m[0]	Put it in A
00100 0 0000010000 0	9:	R16S	ABC>>=6	Shift and store it
00101 0 0000000000 1	10:	TOL	w[0]=AB; ABC=0	so that it becomes the senior 5 bits of m[0] m[1] is now zero
01000 0 000000010 0	11:	T25	m[2]=rdch()	Put next ch in m[2]
11100 0 0000000010 0	12:	A25	A+=m[2]	Put ch in A
01100 0 0000000101 0	13:	S5S	A-=m[5]	A=ch-10
00011 0 0000010101 0	14:	E21S	if A>=0 goto 21	Jump to 21, if ch>=10
00101 0 0000000011 0	15:	T3S	m[3]=A: ABC=0	Clear A. m[3] is junk
11111 0 0000000001 0	16:	V1S	AB+=m[1]*R	A = m[1]*(10<<11)
11001 0 0000001000 0	17:	L8S	A<<=5	Shift 5 more places
11100 0 000000010 0	18:	A2S	A+=m[2]	Add the new digit
00101 0 000000001 0	19:	T1S	m[1]=A; ABC=0	Store back in m[1]
00011 0 0000001011 0	20:	E11S	goto 11	Repeat from 11
				A=2, if ch='S'(=12)
				A=15, if ch='L'(=25)
00100 0 0000000100 0	21:	R4S	ABC>>=4	lenbit=0, if ch='S'
				lenbit=1, if ch='L'
11100 0 0000000001 0	22:	A1S	A+=m[1]	Add in the address
11001 0 0000000000 1	23:	LOL	ABC<<=1	Shift to correct position
11100 0 000000000 0	24:	AOS	A+=m[0]	Add in the operation fiel
00101 0 0000011111 0	25:	T31S	m[31]= A; ABC=0	Store the order in next location
11100 0 0000011001 0	26:	A25S	A+=m[25]	Increment the address field of m[25]
11100 0 0000000100 0	27:	A4S	A+=m[4]	m[4] holds 2
00111 0 0000011001 0	28:	U25S	m[25]=A	Update m[25]
01100 0 0000011111 0	29:	S31S	A-=m[31]	Jump to 6, if there are
11011 0 0000000110 0	30:	G6S	if A<0 goto 6	more orders to load

bit was unused, the following 10 bits was the address field and the last bit specified (where appropriate) whether the order used 17 or 35-bit operands. The instruction and pattern 001010000000000 which is also 10 shifted left 11 places. The instruction instruction at 2 (TOS) assembles into exactly this bit patterns, so is used both as data and as an instruction to clear *m*[0]. The instruction at 3 skips to location 6 over the instructions at 4 and 5 that assembles as the 17-bit constants 2 and 10, respectively.

The main assembly loop starts at 6, leaving locations m[0] to m[5] available as variables and constants in the program. They are used as follows:

- m[0] uses include holding the first character of an order,
- $m[1] \\ m[2]$
- m[3]
- uses include holding the first Caracter of an order, used to hold the address field of the current order, initially 001010...0 as discussed above but also used for characters other than the first of an order, used as a junk register when the instruction at 15 clears ABC, the constant 2 used at 27 to add one to an address field, the constant 10 used to check for the end of address digits.
- m[4] m[5]

of the program is in m[32].



M V Wilkes and WA Renwick

Order bit pattern	Loc	Order	Meaning
00101 0 0001111011 0 00011 0 0001010100 0	31: 32:	T123S E84S	m[123]=A; ABC=0 goto 84
00000 0 000000000 0	33:	PS	data 0
	34:	PS	data U
00000 0 1111000000 0	36:	P100005 P1000S	data 10000<<1
00000 0 0001100100 0	32.	P1005 P105	data 100KCI
00000 0 0000000001 0	39:	P1S	data 1<<1
00001 0 000000000 0	40:	QS	data 1<<12
01011 0 0000000000 0	41:	#S	data 11<<12
11100 0 0000101000 0	42:	A40S	
10100 0 000000000 0	43:	!S	data 20<<12
10010 0 000000000 0	45.	85	data 186612
01001 0 0000101011 0	46:	0435	wr(m[43])
01001 0 0000100001 0	47:	0335	wr(m[33])
00000 0 000000000 0	48:	PS	data O
11100 0 0000101110 0	49:	A46S	A+=m[46]
00101 0 0001000001 0	50:	T65S	m[65]=A; ABC=0
00101 0 0010000001 0	51:	T1295	m[129]=A: ABC=0
11100 0 0000100011 0	52:	A35S	A+=m[35]
00101 0 0000100010 0	53:	T34S	m[34]=A; ABC=0
00011 0 0000111101 0	54:	E61S	goto 61
00101 0 0000110000 0	55:	T48S	m[48]=A; ABC=0
11100 0 0000101111 0	56:	A47S	A +=m[47]
00101 0 0001000001 0	57:	T65S	m[65]=A; ABC=0
11100 0 0000100001 0	58:	A33S	A+=m[33]
11100 0 0000101000 0	59:	A405 7220	A+=m[40]
00101 0 0000100001 0	60:	1335	m[33]-R, ABC-0
11100 0 0000110000 0	61:	A48S	A+=m[48]; ABC=0
00011 0 0000110111 0	63:	E55S	A==m[34] if A>=0 goto 55
11100 0 0000100010 0	64.	4345	4+=m[34]
00000 0 000000000 0	65:	PS	data 0
00101 0 0000110000 0	66:	T48S	m[48]=A; ABC=0
00101 0 0000100001 0	67:	T33S	m[33]=A; ABC=0
11100 0 0000110100 0	68:	A52S	A+=m[52]
11100 0 0000000100 0	69:	A4S	A+=m[4]
01110 0 0000101010 0	70:	052S \$42\$	m[52]=A
11011 0 0000110011 0	72:	G51S	if A<0 goto 51
11100 0 0001110101 0	73:	A117S	A+=m[117]
00101 0 0000110100 0	74:	T52S	m[52]=A; ABC=0
00000 0 000000000 0	75:	PS	data O



The corrected tape segments etched on the Tea Room glass panels

Comment

The required first word Jump to start

For the next decimal digit For the current power of ter

The table of 16-bit powers of ten

00001 in MS 5 bits, used to form digits Figure shift character End limit for values placed in m[52] Space character Line feed character Carriage return charac Write a space Write a digit The number to print

Print subroutine entry point Put 043S in m[65]

Clear A A is next power of ten. m[52] cycles through A358, A368, A378, A38S and A39S Store it in m[34]

Store value to be printed

Store instruction 033S in m[65] Increment the decimal digit held in the MS 5 bits of m[33]

Get value to print Subtract a power of 10 Repeat, if positive

Add back the power of 10 This is replaced by either 0438 to write a space, or 0338 to write a digit Set the value to print Set digit to 0 Increment the address field of the instruction in an (52) in m[52]Compare with A40S and Repeat, if more digits

Put A35S back in m[52] To hold the return jump instruction which is E95S, E110S or E118S

00000	0	000000000	0	76:	PS	data O	Holds x
00000	0	000000000	0	77:	PS	data 0	Holds x^2
00000	0	0000000000	0	78:	PS	data 0	Holds previous x^2
00000	0	0000000000	0	79.	PS	data 0	Holds Δr^2
00000	v	0000000000	0	<i>,,,</i>	15	data 0	
00011	0	0001101110	0	80:	E110S	goto 110	Order to place in m[52]
00011	0	0001110110	0	81:	E118S	goto 118	Order to place in m[52]
00000	0	0001100100	0	82:	P100S	data 100<<1	End limit for x
00011	0	0001011111	0	83:	E95S	goto 95	Order to place in m[52]
01001	~	0000101001	~	0.4	0440	([41])	White Games all ife
01001	U	00001010001	0	84:	0415	wr(m[41])	write ngure shift
00101	0	0010000001	0	85:	T1295	m[129]=A: ABC=0	Start of main loop
01001	ŏ	0000101100	ŏ	86:	0445	wr(m[44])	Write line feed
01001	ō	0000101101	0	87:	0455	wr(m[45])	Write carriage return
11100	õ	0001001100	ō	88:	A765	A+=m 76 : ABC=0	Get x
11100	ō	0000000100	0	89:	A4S	A+=m[4]	Increment it
00111	0	0001001100	0	90-	11765	m[76]=A	and store it back in x
00101	ŏ	0000110000	ŏ	91:	T485	m 48 =A: ABC=0	Put it also in $m[48]$
	-		-				for printing
11100	0	0001010011	0	92:	A83S	A+=m[83]	Put return jump E95S
00101	0	0001001011	0	93:	T75S	m[75]=A; ABC=0	into $m[75]$
00011	0	0000110001	0	94:	E49S	goto 49	Enter the print subroutine
01001	0	0000101011	0	95:	0435	wr(m[43])	Write a space
01001	0	0000101011	0	96:	0435	wr(m[43])	Write a space
10101	0	0001001100	0	97:	H76S	R=m[76]	Multiply x by
11111	0	0001001100	0	98:	V76S	ABC+=m[76]*RS	itself and
11001	0	0001000000	0	99:	L64S	ABC<<8	re-position
11001	0	0000100000	0	100:	L32S	ABC<<7	the result
00111	0	0001001101	0	101:	U77S	m[77]=A	Store in location for x^2
01100	ō	0001001110	ō	102:	S78S	A-=m[78]	Subtract the previous value
00101	0	0001001111	0	103:	T795	m[79]=A: ABC=0	and store the new Δx^2
11100	ŏ	0001001101	ŏ	104:	A77S	A+=m[77]	Update variable holding
00111	0	0001001110	0	105:	U78S	m[78]=A	the previous x^2
00101	0	0000110000	0	106.	T485	m[48]=4 · ABC=0	Put r ²
00101	č	0000110000	Ŭ.	100.	1100		in $m[48]$ for printing
11100	0	0001010000	0	107:	A80S	A+=m[80]	Put return jump E110S
00101	0	0001001011	0	108:	T75S	m[75]=A; ABC=0	into $m[75]$
00011	0	0000110001	0	109:	E49S	goto 49	Enter the print subroutine
						<pre>// //////////////////////////////////</pre>	
01001	0	0000101011	0	110:	0438	wr(m[43])	Write a space
01001	0	0000101011	0	111:	0435	mr(m[43])	Write a space
11100	0	0001001111	0	112:	A79S	A+=m[79]	Get Δx^2
00101	0	0000110000	0	113:	T48S	m[48]=A; ABC=0	Put it in m[48] for printin
11100	0	0001010001	0	114:	A81S	A+=m[81]	Put return jump E118S
00101	0	0001001011	0	115:	T75S	m[75]=A; ABC=0	into m[75]
00011	0	0000110001	0	116:	E49S	goto 49	Enter the print subroutine
11100	0	0000100011	0	117.	1250	A+=[25]	Order to place in an [52]
11100	U	0000100011	0	11/:	A305	A+=m[35]	Order to place in m[52]
11100	0	0001001100	0	118:	A76S	A+=m[76]	Get x
01100	0	0001010010	0	119:	S82S	A-=m[82]	Subtract the end limit (=100
11011	0	0001010101	0	120:	G85S	if A <o 85<="" goto="" td=""><td>Repeat, if more to do</td></o>	Repeat, if more to do
01001	Ō	0000101001	0	121:	041S	wr(m[41])	Write figure shift
01101	0	0000000000	0	122:	ZS	Stop	Stop the machine
						-	· · · · · · · · · · · · · · · · · · ·

The Green Door

The green door on your left was the Corn Exchange Street entrance to the Mathematical Laboratory where EDSAC was built. By convention, the brass plaque on this door holds the engraved names of those retired members of the Laboratory who used the door in its original location.

Links

http://www.dcs.warwick.ac.uk/~edsac/ This links to Martin Campbell-Kelly's excellent EDSAC simulator and related documents

http://www.cl.cam.ac.uk/U0CCL/misc/EDSAC99 This links to pages relating to the celebration, held in Cambridge in April 1999, of the 50th anniversary of the EDSAC1 Computer.

http://www.cl.cam.ac.uk/~mr/Edsac.html

Vorve: c1. cam.ac.uk/ mr/paskc.tram. This links to a shell based EDSAC simulator that runs on Pentium based Linux systems. It was designed to be educational having a built-in interactive debugger allowing single step execution, the setting of breakpoints and convenient inspection and setting of memory and register values. It can be used to explore the execution of the programs described in this poster. This simulator also appears as a demonstration program in the Cintcode entry. BCPL system (http://www.cl.cam.ac.uk/~mr/BCPL.html).

http://www.cl.cam.ac.uk/~mr/edsacposter.pdf This is a PDF version of this poster on two A4 pages AnS SnS EnS GnS LnS OnS TnS UnS XnS ZnS

Acc += Mem[n] Acc -= Mem[n] if Acc >= 0 goto n if Acc < 0 goto n leftshift output Mem[n] = Acc; Acc = 0Mem[n] = AccNo operation Stop the machine and ring a bell



Projects EDSAC History EDSAC News EDSAC in the news **Project Aims Project Organisation** Recreating EDSAC Helping the Project Project Videos EDSAC Remembered

Harwell Dekatron / WITCH ICL 2966 The Colossus Rebuild Robinson Transputer Project Block H ICT 1301 / Flossie

us Supp	ort
Proje	cts
e Lates	t News
Easte	er Bytes
Easte	er Byte

Δ





The EDSAC Replica Project aims to reconstruct one of the most important early British digital computers.

Designed in 1947 by a team lead by Maurice Wilkes, the original EDSAC computer operated for almost 10 years, starting from its first successful program run on 6th May 1949, at the Cambridge University Mathematical Laboratory.

We are now initially targeting to have a working reconstruction of EDSAC as it was in 1951 (when it was in everyday service at the University of Cambridge Mathematical Laboratory) operational by late 2015.

The EDSAC Replica Project is a registered charity and is affiliated to the UK's Computer Conservation Society.

Guided Tours on Tuesdays. Groups any day by appointment. How to book »



http://nhiro.org/learn_language/EDSAC-on-browser.html EDSAC on browser

Back to 'EDSAC on browser' project. Copyright (C) 2012 NISHIO Hirokazu. GPLv3. Short guide: Click 'Load source' and 'Run'. You can click memory view on the left column. Click 'Source' tab and click 'Load Wada Sieve' button. It is very beautiful program of Sieve of Eratosthenes by Eiiti Wada. Back to 'Machine' tab, click 'Reset', 'Load source' and 'Run'.

Machine	Source				
SCR = 0000				Tank #0 (0-31	.)
ABC: [000000		00000000	000000000 0		- - -
ABC = 0, AB					
RS: [00000000000000 0 00000000000000000] RS = 0, R = 0					
Run Step					
Input tape:			Load source		
Switch to Initial	Orders 2				
0000: [0010	01 0 000000000	0 01 [TS]	m[0] = A: ABC	= 0 Output:	
0001: [1010	01 0 00000000	.0 0] [H25	R = I	m[2]	
0002: [0010	01 0 00000000	0 0] [TS]	m[0] = A; ABC	= 0	
0003: [000]	11 0 00000011	0 0] [E65] if A >= 0 got	to 6	
0004: [0000	00 0 00000000	1 0] [P18	5]		
0005: [0000	00 0 00000010	1 0] [P55]		
0006: [0010	01 0 00000000	0 0] [TS]	m[0] = A; ABC	= 0	
0007: [0100	00 0 00000000	0 0] [IS]	m[0] = rea	ad()	
0008: [1110	00 0 000000000	0 0] [AS]	A += 1	n[0]	



"Hi" on the EDSAC / Initial Orders I

T44S	31		T _end+1	mark end o
E38S	32		E _start	jump to be
*S	33	lshift	*	letter shi
HS	34	_H	Н	letter H
IS	35	_I	I	letter I
&S	36	lf	&	LF - line
@S	37	cr	6	CR – carri
033S	38	_start	0 lshift	prepare fo
034S	39		О_Н	print H
035S	40		0 _I	print I
036S	41		0 lf	print lf
037S	42		0 cr	print cr
ZS	43	_end	Z	end of pro

T44SE38S*SHSIS&S@S033S034S035S036S037SZS

of program eginning of program ft

feed character age return character or printing lettersn

gram

written in a "primitive" 1949-like style by Olve Maudal, Monday, April 20, 2015

I pretended I was a student, who had won a single chance to run my program on this precious computer.

The program did actually ran on the very first attempt!

T123S	31	T L end	mark end of program
E60S	32	E L start	iump to the beginning of program
#S	33 FS	#	figure shift
*S	34 LS	*	letter shift
&S	35 LF	&	linefeed character
@S	36 CR	6	carriage return character
P100S	37 100	P 100	constant 100
P105	38 10	P 10	constant 10
P5S	39 5	P 5	constant 5
P3S	40 3	P 3	constant 3
P1S	41 1	P 1	constant 1
05	42 '1'	0	constant figure 1
PS	43 '0'	P	constant figure 0
BS	44 B	B	constant letter B
FS	45 F	F	constant letter F
TS	46 T	т	constant letter I
us	47 11	Ū.	constant letter U
75	48 7	7	constant letter 7
25 PS	40 _2 49 dummy	P	used to flush and reset the accumulator
P1S	50 cnt	P 1	counter current number to be considered will be increased
DS	50 _circ	D D	number to be printed, possible if counter is med 2 or med 5
PS	52 d	P	digit to be printed
0345	53 L nevt	0 15	output IS prepare for printing letters
0255	55 L_Hext	0_L5	output L5, prepare for printing tetters
0353	55		output CP, carriage return
T100	56		reset Arc
1433	50	A cot	load Acc with ont
A305	51		increase Acc
A415	30 50	^ _⊥ T	niciease Acc
1505	22		Score Acc mild _cmt, reset Acc
A505	o⊍ ∟_start	A _COT	LOAD ALC WITH _CHT (WE KNOW THAT ACC INITIALLY IS 0)
0515	C2 +	ບ_num ເ	centatively set number to be printed
5405	62 L_try⊦izz	S_3	
E62S	63	⊾ L_tryFizz	LOOP UNTIL ACC < 0
A40S	64	A _3	add 3, restore previous value
S41S	65	5_1	subtract 1, to check it Acc was 0
E73S	66	E L_notFizz	jump it Acc was not 0, ie number was not divisable by 3
T51S	67	T _num	set _num to negative value, flag that no value should be printed
034S	68	0 _LS	prepare printing letters
045S	69	0_F	output F
046S	70	0 _I	output I
048S	71	0 _Z	output Z
048S	72	0 _Z	output Z
T49S	73 L_notFizz	T _dummy	reset Acc
A50S	74	A _cnt	load Acc with _cnt
\$39\$	75 L_Buzz	S _5	subtract 5
E75S	76	E L_Buzz	loop until Acc < 0
A39S	77	A _5	add 5, restore previous value
\$41\$	78	S _1	subtract 1, to check if Acc was 0
E86S	79	E L_notBuzz	jump if Acc was not 0, ie number was not divisable by 5
T51S	80	T _num	set _num to negative value, flag that no value should be printed
034S	81	0 _LS	prepare printing letters
044S	82	0 _B	output B
047S	83	0 _U	output U
048S	84	0 _Z	output Z
048S	85	0 _Z	output Z
T49S	86 L_notBuzz	T _dummy	reset Acc
A51S	87	A _num	load _num to check number to be printed
G53S	88	G L_next	goto next iteration if _num is negative
0335	89 L_printNum	0 _FS	prepare for printing numbers
T49S	90	T _dummy	reset Acc
A50S	91	A _cnt	load counter
\$37\$	92	S _100	subtract 100, check if we should stop
G98S	93	G L_not100	jump if not 100 yet
042S	94	0 _'1'	output 1
043S	95	0 _'0'	output 0
043S	96	0 _'0'	output 0
ZS	97	Z	end the program
T49S	98 L_not100	T _dummy	reset Acc
T52S	99	T _d	reset digit
A50S	100	A _cnt	load counter
\$38\$	101 L_count10s	S _10	subtract 10
G109S	102	G L_print10s	goto print 10s if Acc < 0
T51S	103	T _num	store number
A52S	104	A _d	load digit
A41S	105	A _1	increase digit
T52S	106	T _d	store digit
A51S	107	A _num	load number
E101S	108	E L_count10s	loop unconditionally
T49S	109 L_print10s	T _dummy	reset Acc
A52S		A d	load digit
	110	A _u	
\$41S	110 111	S _1	decrease digit by 1
S41S G117S	110 111 112	S _1 G L_1	decrease digit by 1 if negative (digit was 0), skip printing of tens digits
S41S G117S A41S	110 111 112 113	S _1 G L_1 A _1	decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1
S41S G117S A41S L512S	110 111 112 113 114	S _1 G L_1 A _1 L 2^(11-2)	decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11. create a printable figure
S41S G117S A41S L512S T52S	110 111 112 113 114 115	S _1 G L_1 A _1 L 2^(11-2) T d	decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save printable figure
S41S G117S A41S L512S T52S 052S	110 111 112 113 114 115 116	S _1 G L_1 A _1 L 2^(11-2) T _d O d	decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save printable figure print figure / digit
S41S G117S A41S L512S T52S 052S T49S	110 111 112 113 114 115 116 117 L 1:	S _1 G L_1 A _1 L 2^(11-2) T _d O _d T _dummy	<pre>decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save printable figure print figure / digit reset Acc</pre>
S41S G117S A41S L512S T52S 052S T49S A51S	110 111 112 113 114 115 116 117 L_1: 118	S _1 G L_1 A _1 L 2^(11-2) T _d O _d T _dummy A _pum	decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save printable figure print figure / digit reset Acc load number
S41S G117S A41S L512S T52S 052S T49S A51S L512S	110 111 112 113 114 115 116 117 L_1: 118 119	S _1 G L_1 A _1 L 2^(11-2) T _d O _d T _dummy A _num L 2^(11-2)	<pre>decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save printable figure print figure / digit reset Acc load number Acc << 11, create a printable figure</pre>
S41S G1175 A41S L512S T52S 052S T49S A51S L512S T52S	110 111 112 113 114 115 116 117 L_1: 118 119 120	S _1 G L_1 A _1 L 2^(11-2) T _d O _d T _dummy A _num L 2^(11-2) T d	decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save printable figure print figure / digit reset Acc load number Acc << 11, create a printable figure save printable figure
S41S G117S A41S L512S T52S 052S T49S A51S L512S T52S 052S	110 111 112 113 114 115 116 117 L_1: 118 119 120 121	<pre>X _0 S _1 G L_1 A _1 L 2^(11-2) T _d O _d T _dummy A _num L 2^(11-2) T _d O _d</pre>	<pre>decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save printable figure print figure / digit reset Acc load number Acc << 11, create a printable figure save printable figure print figure / digit</pre>
S41S G117S A41S L512S O52S T49S A51S L512S T52S O52S E53S	110 111 112 113 114 115 116 117 L_1: 118 119 120 121 122	<pre>X _u S _u G L_1 A _1 L 2^(11-2) T _d O _d T _dummy A _num L 2^(11-2) T _d O _d E L next</pre>	<pre>decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save printable figure print figure / digit reset Acc Load number Acc << 11, create a printable figure save printable figure print figure / digit unconditional jump</pre>

T123S	31	T L_end	mark end of pro
E60S	32	E L_start	jump to the beg
#S	33 _FS	#	figure shift
*S	34 _LS	*	letter shift
&S	35 _LF	&	linefeed charad
@S	36 _CR	0	carriage retur
P100S	37 _100	P 100	constant 100
P10S	38 _10	P 10	constant 10
P5S	39 _5	P 5	constant 5
P3S	40 _3	P 3	constant 3
P1S	41 _1	P 1	constant 1
QS	42 _'1'	Q	constant figure
PS	43 _'0'	Р	constant figure
BS	44 _B	В	constant letter
FS	45 _F	F	constant letter
IS	46 _I	I	constant letter
US	47 _U	U	constant letter
ZS	48 _Z	Z	constant letter
PS	49 _dummy	Р	used to flush a
P1S	50 _cnt	P 1	counter, curren
PS	51 _num	Р	number to be p
PS	52 _d	Р	digit to be pr

ogram ginning of program

acter n character

gure 1 gure 0 tter B tter F tter I tter U tter Z sh and reset the accumulator rrent number to be considered, will be increased e printed, negative if counter is mod 3 or mod 5 printed

T122C	21	T I and	mark and of program
11233	31	T L_enu	inark end of program
E003	32 55	E L_Start	jump to the beginning of program
#5	33 _FS	#	Tigure shift
*5	34 _LS	*	letter shift
&S	35 _LF	&	linefeed character
@S	36 _CR	0	carriage return character
P100S	37 _100	P 100	constant 100
P10S	38 _10	P 10	constant 10
P5S	39 _5	P 5	constant 5
P3S	40 _3	P 3	constant 3
P1S	41 _1	P 1	constant 1
QS	42 _'1'	Q	constant figure 1
PS	43 _'0'	Р	constant figure 0
BS	44 _B	В	constant letter B
FS	45 _F	F	constant letter F
IS	46 _I	I	constant letter I
US	47 _U	U	constant letter U
ZS	48 _Z	Z	constant letter Z
PS	49 _dummy	Р	used to flush and reset the accumulator
P1S	50 _cnt	P 1	counter, current number to be considered, will be increased
PS	51 _num	Р	number to be printed, negative if counter is mod 3 or mod 5
PS	52 _d	Р	digit to be printed
034S	53 L_next	0 _LS	output LS, prepare for printing letters
035S	54	0 _LF	output LF, linefeed
036S	55	0 _CR	output CR, carriage return
T49S	56	T _dummy	reset Acc
A50S	57	A _cnt	load Acc with _cnt
A41S	58	A _1	increase Acc
T50S	59	T _cnt	store Acc into _cnt, reset Acc
A50S	60 L_start	A _cnt	load Acc with _cnt (we know that Acc initially is 0)
U51S	61	U_num	tentatively set number to be printed
\$40\$	62 L_tryFizz	S _3	subtract 3
E62S	63	E L trvFizz	loop until Acc < 0
A40S	64	A 3	add 3. restore previous value
S41S	65	S 1	subtract 1. to check if Acc was 0
E735	66	F L notFizz	jump if Acc was not 0, ie number was not divisable by 3
T515	67	T num	set num to negative value. flag that no value should be printed
0345	68	0 15	prenare printing letters
0455	69	0 E	outout F
0465	70	0 T	
0485	70	0 7	output 7
0485	72	0 7	output Z
T495	73 L notFizz		reset Acc
A50S	74	A cnt	load Acc with cnt
\$395	75 L Buzz	S 5	subtract 5
E75S	76	E L Buzz	loop until Acc < 0
A39S	77	A _5	add 5, restore previous value
S41S	78	S _1	subtract 1, to check if Acc was 0
	70	E potRuzz	jump if Acc was not 0, ie number was not divisable by 5
E86S	19	L L_HOLDUZZ	
E86S T51S	80	T_num	set _num to negative value, flag that no value should be printed
E86S T51S 034S	80 81	T _num O _LS	<pre>set _num to negative value, flag that no value should be printed prepare printing letters</pre>
E86S T51S 034S 044S	80 81 82	T _num O _LS O _B	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B</pre>
E86S T51S 034S 044S 047S	79 80 81 82 83	T _num O _LS O _B O _U	set _num to negative value, flag that no value should be printed prepare printing letters output B output U
E86S T51S 034S 044S 047S 048S	79 80 81 82 83 84	T _num O _LS O _B O _U O _Z	set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z
E86S T51S 034S 044S 047S 048S 048S	80 81 82 83 84 85	Tnum O _LS O _B O _U O _Z O _Z	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z</pre>
E86S T51S 034S 044S 047S 048S 048S 048S T49S	79 80 81 82 83 84 85 85 86 L_notBuzz	C L_INCROUZZ T _num O _LS O _U O _U O _Z O _Z T _dummy	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc</pre>
E86S T51S 034S 044S 047S 048S 048S T49S A51S	80 81 82 83 84 85 86 L_notBuzz 87	Tnum O _LS O _B O _U O _Z O _Z T _dummy Anum	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed</pre>
E86S T51S 034S 044S 047S 048S 048S T49S A51S G53S	80 81 82 83 84 85 86 L_notBuzz 87 88	Tnum O _LS O _B O _U O _Z T _dummy Anum G Lnext	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative</pre>
E86S T51S 034S 044S 047S 048S 048S T49S A51S G53S 033S	79 80 81 82 83 84 85 85 86 L_notBuzz 87 88 89 L_printNum	C T OLS OB OU OZ OZ Tdummy Anum G Lnext OFS	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers</pre>
E86S T51S 034S 044S 047S 048S 048S T49S A51S G53S 033S T49S	79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90	C L_10CBU22 T_num O _LS O _B O _U O _Z O _Z T_dummy A _num G L_next O _FS T_dummy	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc</pre>
E86S T51S 034S 044S 047S 048S T49S A51S G53S 033S T49S A50S	79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91	L L_10CBU22 T_num O _LS O _B O _U O _Z T_dummy A _num G L_next O _FS T_dummy A _cnt	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter</pre>
E86S T51S 044S 044S 047S 048S 048S T49S A51S G53S 033S T49S A50S S37S	<pre>79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92</pre>	<pre>L L_INCEUZ2 Tnum 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy Anum G L_next 0 _FS T _dummy Acnt S100</pre>	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop</pre>
E86S T51S 044S 044S 047S 048S T49S A51S G53S T49S A50S S37S G98S	<pre>79 80 81 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93</pre>	<pre>L L_INCEUZ2 T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100</pre>	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet</pre>
E865 T51S 034S 044S 047S 048S T49S A51S G53S 033S T49S A50S S37S G98S 042S	79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94	L L_10CBU22 T_num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1'	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1</pre>
E86S T51S 034S 044S 047S 048S T49S A51S 653S 033S T49S A50S S37S 698S 042S 043S	<pre>79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95</pre>	<pre>L L_inctu22 L L_inctu22 T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0'</pre>	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0</pre>
E865 T515 0345 0445 0445 0475 0485 0485 T495 A515 G535 0335 T495 A505 S375 G985 0435 0435	<pre>79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96</pre>	L L_10CBU22 T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0'	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0</pre>
E865 T515 0445 0445 0445 0485 0485 0485 T495 A515 G535 0335 T495 A505 S375 G985 0425 0425 0435 ZS	<pre>79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97</pre>	<pre>L L_INCEUZZ L L_INCEUZZ T_UMM O _LS O _B O _U O _Z O _Z T_dummy A _num G L_next O _FS T_dummy A _cnt S _100 G L_not100 O _'1' O _'0' O _'0' Z</pre>	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 end the program</pre>
E865 T51S 034S 044S 044S 048S 048S T49S A51S G53S 033S T49S A50S S37S G98S 042S 042S 043S 2S T49S	79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100	L L_10/GU22 T_num O _LS O _B O _U O _Z O _Z T_dummy A _num G L_next O _FS T_dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T_dummy	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 end the program reset Acc</pre>
E865 T515 0345 0445 0475 0485 T495 A515 G535 0335 T495 A505 S375 G985 S425 0425 0435 Z5 T495 T525	<pre>79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99</pre>	L L_10CBU22 Tnum O _LS O _B O _U O _Z T _dummy Anum G L_next O _FS T _dummy Acnt S _100 G L_not100 O _'1' O _'0' Z _ Zdummy Td	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc reset digit</pre>
E865 T515 0345 0445 0475 0485 T495 A515 G535 0335 C495 A505 S375 G985 0425 0435 0435 Z5 T495 T525 A505	<pre>%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%</pre>	<pre>L L=106022 T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt</pre>	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 end the program reset Acc reset digit load counter</pre>
E865 T515 0345 0445 0445 0475 0485 0485 T495 A515 G535 0335 T495 A505 S375 G985 0425 0435 0435 0435 25 T495 T495 T525 S385	<pre>%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%</pre>	L L=1000222 T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS _100 G L_next O _FS _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc reset digit load counter subtract 10</pre>
E86S T51S 034S 044S 044S 048S 048S T49S A51S 653S 033S T49S S37S 698S 042S 043S 043S 2S T49S T52S X38S 538S 6109S	<pre>%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%</pre>	L L_10CBU22 T_num O _LS O _B O _U O _Z O _Z T_dummy A _num G L_next O _FS T_dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T_dummy T_d A _cnt S_100 G L_print10s	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0</pre>
E86S T51S 034S 044S 047S 048S T49S A51S 033S T49S A50S S37S 042S 043S C42S 043S C43S C43S C43S C43S C43S C43S C43S C	<pre>79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_count10s 102</pre>	L L_10C6022 Tnum O _LS O _B O _U O _Z T _dummy Anum G L_next O _FS T _dummy Acnt S _100 G L_not100 O _'1' O _'0' Z U _'0' Z Z dummy T dummy T dummy T d Acnt S I00 G L_print10s Tnum	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number</pre>
E865 T515 0345 0445 0445 0475 0485 T495 A515 G535 0485 T495 A505 S375 G985 C425 0435 C425 0435 C435 C435 C435 C435 C435 C435 C435 C	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_count10s 102 103 104</pre>	L L=10:0022 T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_not100 O _'1' O _'0' Z J _0''0' Z _dummy T _d A _cnt S _10 G L_not100 O _'1' O _'0' Z _dummy A _cnt S _10 Gnot100 O _'1' Cnot100 O _'1' A _cnt S10 A _cnt A _cnt A _cnt A _cnt A _cnt A _cnt A _cnt A _cnt A _cnt	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit</pre>
E865 T515 0345 0445 0445 0475 0485 0485 T495 A515 G535 0335 T495 A505 S375 G985 0435 C435 C435 C435 C435 T525 A505 S385 G1095 T515 A525 A515 A525 A415	<pre>%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%</pre>	L L_100022 T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_no1100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_printl0s T _num A _cnt S _10 G L_printl0s T _num A _cnt S _10 G L_printl0s T _num	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit</pre>
E865 T515 0445 0445 0445 0445 0485 0485 0485 048	<pre>%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%</pre>	L L_100022 Tnum 0 _LS 0 _B 0 _U 0 _Z 0 _Z Tdummy A _num G L_next 0 _FS Tdummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' Z T _dummy Td A _cnt S _10 G L_not100 0 _'1' 0 _'0' Z T _dummy T _d A _cnt S _10 G L_print10S T _num A _cnt Z T _dummy T _d A _cnt Z T _dummy T _d A _cnt Z T _dummy T _d A _cnt Z T _dummy T _d A _cnt S _10 G L_not100 0 _'1' 0 _'0' Z T _dummy T _d A _cnt Z T _dummy A _cnt A _cnt A _cnt C _0 C	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit</pre>
E865 T515 0345 0445 0445 0475 0485 T495 A515 G535 0335 T495 A505 S375 G985 0425 0435 C425 0435 C435 C435 C435 C435 C435 C435 C435 C	<pre>// 9 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_count10s 102 103 104 105 106 107</pre>	L L_10CBU22 Tnum O _LS O _B O _U O _Z T _dummy Anum G L_next O _FS T _dummy Acnt S _100 G L_not100 O _'1' O _'0' C _'0' Z T _dummy T _d Acnt S _100 G L_print10s Tnum Ad A1 T _d Anum	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit load number</pre>
E865 T515 0345 0445 0475 0485 T495 A515 G535 0485 T495 A505 S375 G985 A505 S375 G425 0435 C425 0435 C425 C435 C435 C435 C435 C435 C435 C435 C43	<pre>// 9 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_count10s 102 103 104 105 106 107 108</pre>	L L_10C6022 Tnum O _LS O _B O _U O _Z T _dummy Anum G L_next O _FS T _dummy Acnt S _100 G L_not100 O _'1' O _'0' Z C C C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z C'0' Z'0' Z C'0' Z'0'	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number loop unconditionally</pre>
E865 T515 0345 0445 0475 0485 T495 A515 G535 0335 T495 A505 S375 G985 0425 0435 0435 C425 A505 S385 G1095 T525 A505 S385 G1095 T525 A415 T525 A415 T525 A415 T525 A515 S355 S355 S355 S355 S355 S355 S355 S	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_count10s 102 103 104 105 106 107 108 109 L_print10s</pre>	<pre>L L_INCRUZZ L L_INCRUZZ T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z _ T _dummy T _d A _cnt S _10 G L_print10s T _num A _cnt S _10 G L_print10s T _num E L_count10s T _dummy</pre>	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print los if Acc < 0 store number load digit increase digit store digit load number loop unconditionally reset Acc</pre>
E86S T51S 044S 044S 044S 044S 047S 048S 048S 048S 048S 043S 043S 043S 043S 043S 043S 043S 043	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_count10s 102 103 104 105 106 107 108 109 L_print10s 110</pre>	L L_INCRUZZ T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_printl0S T _num A _d A _num E L_countl0S T _dummy A _d	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit store digit load number load number load number load number</pre>
E865 T515 0345 0445 0445 0475 0485 T495 A515 G535 0335 T495 A505 S375 0425 0435 C435 C435 C435 C435 C435 C435 C435 C	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_count10s 102 103 104 105 106 107 108 109 L_print10s 110</pre>	<pre>L L_initial2 L L_initial2 L L_initial2 D _LS D _LS D _L D _L D _L D _L D _L D _L D _L D _L</pre>	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number loop unconditionally reset Acc load digit decrease digit by 1</pre>
E86S T51S 034S 044S 047S 048S T49S A51S G53S 048S T49S A50S S37S G98S 042S 043S 043S C42S 043S C42S 043S C42S 043S C42S 043S C42S 043S C52S A50S S38S G10S T51S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S C52S A51S C53S C53S C53S C53S C53S C53S C53S C53	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_count10s 102 103 104 105 106 107 108 109 L_print10s 111 112</pre>	L L_10C6022 T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z _ T _dummy T _d A _cnt S _10 G L_print10s T _num A _cnt S _10 G L_print10s T _num A _cnt S _10 G L_print10s T _num A _cnt S _10 G L_print10s T _dummy A _cnt S _11 G L_print10s T _dummy A _cnt S _11 C _nnt A _cnt C _nnt C _nnt C _nnt C _nnt	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number loop unconditionally reset Acc load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits</pre>
E865 T515 0345 0445 0475 0485 T495 A515 G535 0335 T495 A505 S375 G985 0425 0435 0435 0435 C435 C435 C435 C435 C435 C435 S385 G1095 T525 A505 S385 G1095 T525 A415 T495 A525 S415 C415 S415	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_countl0s 102 103 104 105 106 107 108 109 L_print10s 110 111 112 113</pre>	<pre>L L_10:0022 L L_10:0022 Tnum 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _1 T _d A _1 A _1 C _1 A _1 </pre>	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number loop unconditionally reset Acc load digit if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1</pre>
E865 T515 0345 0445 0445 0445 0475 0485 T495 A515 G535 0335 T495 A505 S375 G985 0435 0435 0435 C435 C435 C435 C435 S155 S385 G1095 T525 A415 T525 A515 E1015 E1495 A525 S415 G1175 A415	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_countl0s 102 103 104 105 106 107 108 109 L_print10s 110 111 112 113 114</pre>	<pre>L L_10/C0222 L L_10/C0222 Tdumm 0 _LS 0 _B 0 _U 0 _Z 0 _Z Tdummy Anum G L_next 0 _FS Tdummy Acnt S100 G L_not100 0 _'1' 0 _'0' Z Tdummy Td Acnt S10 G L_print10s Tnum Ad A1 Td A1 Td Ad S1 G L_1 A1 L2^(11-2)</pre>	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load number load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure</pre>
E865 T515 0345 0445 0445 0445 0475 0485 T495 A515 G535 0335 T495 A505 S375 0425 0435 C425 0435 C425 0435 C435 C435 C435 C435 C435 C525 C415 C525 C415 C525	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 94 90 91 92 93 94 95 96 97 98 L_not100 99 100 L_count10s 102 103 104 105 106 107 108 109 L_print10s 110 112 113 114</pre>	L L_10CBU22 Tnum O _LS O _B O _U O _Z Tdummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_nct100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 G L_print10s T _num A _cnt S _10 G L_print10s T _num A _d A _num E L_count10s T _dummy A _d S _1 G L_1 A _1 L 2 (11-2) T _d	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number loop unconditionally reset Acc load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit. Acc << 11, create a printable figure save printable figure</pre>
E865 T515 0345 0445 0445 0475 0485 T495 A515 G535 0425 0425 0435 0425 0435 0425 0435 C425 0435 C425 0435 C425 C435 C525 A515 E1015 T495 A515 E1015 T495 A515 E1015 T495 A525 S415 C175 A415 E5125 T525 S415 C525	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 9 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_count10s 102 103 104 105 106 107 108 109 L_print10s 111 112 113 114 115 116</pre>	L L_10C6022 T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' C _'0' Z _ T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _num E L_count10s T _dummy A _d S _1 G L_1 A _num E L_count10s T _dummy A _d S _1 G L_1 A _1 L _0 (1-2) T _d A _1 C	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit load number loop unconditionally reset Acc load digit if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save print figure / digit</pre>
E86S T51S 034S 044S 044S 047S 048S T49S A51S 653S 033S 749S A50S S37S 698S 042S 043S 043S 043S 043S 043S 043S 043S 043	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_count10s 102 103 104 105 106 107 108 109 L_print10s 111 112 113 114 115 116 117 L_1:</pre>	L L=100022 T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z _ dummy T _d A _cnt S _100 G L_not100 O _'1' O _'0' Z _ dummy T _d A _cnt S _10 G L_print108 T _dummy T _d A _cnt S _10 G L_print108 T _dummy A _d A _1 T _dummy A _d S _1 G L_1 A _1 L 2^(11-2) T _d O _d T _dummy	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number loop unconditionally reset Acc load digit if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure print figure / digit reset Acc</pre>
E865 T515 0345 0445 0445 0445 0475 0485 T495 A515 G535 0435 0435 0435 0435 0435 0435 0435 0	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_countl0s 102 103 104 105 106 107 108 109 L_printl0s 110 111 112 113 114 115 116 117 L_1: 118</pre>	<pre>L L_10/C0222 L L_10/C0222 TUUM 0 _LS 0 _B 0 _U 0 _Z 0 _Z Tdummy Anum G L_next 0 _FS Tdummy Acnt S _100 G L_no1100 0 _'1' 0 _'0' Z Tdummy Td Acnt S _10 G L_print10s Tnum E L_count10s Td Ad A1 Td Ad A1 Td Ad S1 G L_1 A1 L 2^(11-2) Td UMMY Anum </pre>	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure print figure / digit reset Acc load number</pre>
E865 T515 O445 O445 O445 O445 O445 O445 A505 S375 G985 O435 C435 O435 C435 O435 C435 C435 C435 C435 C435 C435 C435 C	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 L_countl0s 102 103 104 105 106 107 108 109 L_print10s 110 111 112 113 114 115 116 117 L_1: 118 119</pre>	L L_100022 T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _cnt S _10 G L_print10s T _dummy A _d A _1 T _d A _num E L_count10s T _dummy A _d S _1 G L_1 A _d S _1 G L_1 A _d M _d S _1 C _1 Z T _dummy A _d A _d S _1 G L_1 A _d M _d M _d S _1 C _1 Z T _dummy A _d A _d	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit load number loop unconditionally reset Acc load digit if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure print figure / digit reset Acc load number Acc << 11, create a printable figure</pre>
E865 T515 0345 0445 0445 0475 0485 T495 A515 G535 0425 0425 0435 0425 0435 0425 0435 0435 C425 0435 C425 0435 C425 C435 C435 C525 C525 T495 A515 C5125 T495 A515 C5125 C525	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 9 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_countl0s 102 103 104 105 106 107 108 109 L_print10s 111 112 113 114 115 116 117 L_1: 118 119 120</pre>	L L_10C6022 T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' C _'0' C _'0' Z T _dummy T _d A _cnt S _100 G L_print10s T _num A _d A _1 T _d A _num E L_count10s T _dummy A _d A _1 C _1	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number loop unconditionally reset Acc load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save printable figure save printable figure</pre>
E86S T51S 034S 044S 047S 044S 047S 048S T49S A51S G53S 042S 043S 043S 043S 043S 043S 043S 043S 043	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_count10s 102 103 104 105 106 107 108 109 L_print10s 111 112 113 114 115 116 117 L_11: 118 119 120 121</pre>	L L=100022 T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' C _'0' Z _ dummy T _d A _cnt S _10 G L_not100 O _'1' O _'0' Z _ dummy T _d A _cnt S _10 G L_print108 T _dummy A _d A _num E L_count108 T _dummy A _d A _1 T _dummy A _d A _1 C _10 C _10	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load number load digit if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save printable figure print figure / digit save printable figure print figure / digit</pre>
E865 T515 0345 0445 0445 0445 0475 0485 T495 A515 G535 0425 0435 0435 0435 0435 0435 0435 0435 043	<pre>//9 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 101 L_countl0s 102 103 104 105 106 107 108 109 L_printl0s 110 111 122 113 114 115 116 117 L_1: 118 119 120 121 122</pre>	L L-INCEGUZZ T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z _ T _dummy T _d A _cnt S _10 G L_not100 O _'1' O _'0' Z _ T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _1 T _d A _num E L_count10s T _dummy A _d S _1 G L_1 A _1 L 2^(11-2) T _d O _d E L_next	<pre>set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits rester digit if negative (digit measing with 1 Acc << 11, create a printable figure print figure / digit nuconditional jump</pre>

034S	53 L_next	0 _LS
035S	54	0 _LF
036S	55	0 _CR
T49S	56	T _dummy
A50S	57	A _cnt
A41S	58	A _1
T50S	59	T _cnt
A50S	60 L_start	A _cnt
U51S	61	U _num
S40S	62 L_tryFizz	S_3
E62S	63	E L_tryFizz
A40S	64	A _3
S41S	65	S _1
E73S	66	E L_notFizz
T51S	67	T _num
034S	68	0 _LS
045S	69	0 _F
046S	70	0 _I
048S	71	0 _Z
048S	72	0 _Z

output LF, linefeed output CR, carriage return reset Acc load Acc with _cnt increase Acc subtract 3 loop until Acc < 0prepare printing letters output F output I output Z output Z

```
output LS, prepare for printing letters
store Acc into _cnt, reset Acc
load Acc with _cnt (we know that Acc initially is 0)
tentatively set number to be printed
add 3, restore previous value
subtract 1, to check if Acc was 0
jump if Acc was not 0, ie number was not divisable by 3
set _num to negative value, flag that no value should be printed
```

T1000		- · · ·	
11235	31	I L_end	mark end of program
E60S	32	E L_start	jump to the beginning of program
#S	33 _FS	#	figure shift
*S	34 _LS	*	letter shift
&S	35 _LF	&	linefeed character
@S	36 _CR	0	carriage return character
P100S	37 _100	P 100	constant 100
P10S	38 _10	P 10	constant 10
P5S	39 _5	P 5	constant 5
P3S	40 3	P 3	constant 3
P1S	41 1	P 1	constant 1
05	42 '1'	0	constant figure 1
PS	43 '0'	P	constant figure 0
RS	44 B	R	constant letter B
FS	45 F	F	constant letter F
TS	46 T	т	constant letter T
15	40 _1	1	constant letter I
70	40 7	7	constant letter 7
25	48 _Z	2	constant letter Z
P5	49 _dummy	P	used to flush and reset the accumulator
PIS	50 _cnt	PI	counter, current number to be considered, will be increased
PS	51 _num	P	number to be printed, negative if counter is mod 3 or mod 5
PS	52 _d	P	digit to be printed
034S	53 L_next	0 _LS	output LS, prepare for printing letters
035S	54	0 _LF	output LF, linefeed
036S	55	0 _CR	output CR, carriage return
T49S	56	T _dummy	reset Acc
A50S	57	A _cnt	load Acc with _cnt
A41S	58	A _1	increase Acc
T50S	59	T _cnt	store Acc into _cnt, reset Acc
A50S	60 L_start	A _cnt	load Acc with _cnt (we know that Acc initially is 0)
U51S	61	U _num	tentatively set number to be printed
S40S	62 L_tryFizz	S _3	subtract 3
E62S	63	E L_tryFizz	loop until Acc < 0
A405	64	A 3	add 3, restore previous value
\$41S	65	S 1	subtract 1, to check if Acc was 0
F735	66	F L notFizz	jump if Acc was not 0, ie number was not divisable by 3
T51S	67	T num	set num to negative value flag that no value should be printed
0345	68	0 15	nrenare printing letters
0455	60	0_L3	outout E
0455	70	0_F	
0403	70	0_1	output I
0465	71	0_2	
0485	72	U_2	
1495	73 L_NOTFIZZ	I _dummy	reset Acc
A505	74	A _Cht	LOAD ACC WITH _CHT
2282	75 L_BUZZ	5 5	SUDTRACT 5
EZEC	70		lass with the CO
E75S	76	E L_Buzz	loop until Acc < 0
E75S A39S	76 77	E L_Buzz A _5	loop until Acc < 0 add 5, restore previous value
E75S A39S S41S	76 77 78	E L_Buzz A _5 S _1	loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0
E75S A39S S41S E86S	76 77 78 79	E L_Buzz A _5 S _1 E L_notBuzz	loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5
E75S A39S S41S E86S T51S	76 77 78 79 80	E L_Buzz A _5 S _1 E L_notBuzz T _num	loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed
E75S A39S S41S E86S T51S 034S	76 77 78 79 80 81	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters</pre>
E75S A39S S41S E86S T51S 034S 044S	76 77 78 79 80 81 82	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B</pre>
E75S A39S S41S E86S T51S 034S 044S 047S	76 77 78 79 80 81 82 83	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U</pre>
E75S A39S S41S E86S T51S 034S 044S 047S 048S	76 77 78 79 80 81 82 83 83	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z</pre>
E75S A39S S41S E86S T51S 034S 044S 047S 048S 048S	76 77 78 79 80 81 82 83 83 84 85	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z</pre>
E75S A39S S41S E86S T51S 034S 044S 044S 047S 048S 048S T49S	76 77 78 80 81 82 83 84 85 85 86 L_notBuzz	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc</pre>
E75S A39S S41S E86S T51S 034S 044S 047S 048S 048S T49S A51S	76 77 78 80 81 82 83 84 85 86 L_notBuzz 87	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed</pre>
E75S A39S S41S E86S T51S 034S 044S 044S 047S 048S 048S T49S A51S G53S	76 77 78 80 81 82 83 84 85 86 L_notBuzz 87 88	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative</pre>
E75S A39S S41S E86S T51S 034S 044S 044S 044S 048S T49S A51S G53S 033S	76 77 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers</pre>
E75S A39S S41S E86S T51S 034S 044S 044S 044S 048S T49S A51S G53S 033S T49S	76 77 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc</pre>
E75S A39S S41S E86S T51S 034S 044S 044S 048S T49S A51S G53S 03S T49S A50S	76 77 78 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter</pre>
E75S A39S S41S E86S T51S 034S 047S 048S 047S 048S T49S 653S 033S T49S A50S S37S	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop</pre>
E75S A39S E86S T51S 034S 047S 048S 048S 048S T49S A51S 033S T49S A50S S37S 698S	76 77 78 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 93	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L not100	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet</pre>
E75S A39S S41S E86S T51S 034S 044S 048S 048S 048S T49S A51S 033S T49S A50S S37S G98S 042S	76 77 78 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1'	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1</pre>
E75S A39S S41S E86S T51S 044S 044S 048S 048S T49S A51S 653S 033S T49S S37S 698S 043S	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _ '1' O _0'	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 </pre>
E75S A39S S41S E86S T51S 034S 047S 048S 047S 048S T49S 653S 033S T49S S37S 698S 042S 043S	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' O _'0'	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 </pre>
E75S A39S E86S T51S 034S 047S 048S 048S 048S T49S A50S 653S 033S T49S A50S S37S 698S 042S 043S 043S 043S	76 77 78 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 output 0 output 0</pre>
E75S A39S S41S E86S T51S 034S 047S 048S 048S 048S T49S A51S 033S T49S A50S S37S 033S T49S A50S S37S 042S 042S 043S 043S 22	76 77 78 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L not100	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _Z T _dummy A _cnt S _100 G L_not100 O _'0' Z T _dummy	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc</pre>
E75S A39S S41S E86S T51S 044S 044S 044S 048S 048S 451S 653S 033S T49S S37S 698S 042S 043S 043S 043S 2S T49S 2S	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _ '1' O _'0' O _'0' Z T _dummy T _dummy	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 output 0 end the program reset Acc</pre>
E75S A39S S41S E86S T51S 034S 047S 048S 047S 048S 048S T49S S37S 698S 042S 043S ZS T49S T52S	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc load counter </pre>
E75S A39S S41S E86S T51S 034S 047S 048S 048S 048S 048S T49S 653S 033S T49S S37S 698S 042S 043S 043S 043S 043S 043S 043S 043S 043	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 97 98 L_not100 99 100	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 C _10' C _1	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc reset digit load counter subtract 100</pre>
E75S A39S S41S E86S T51S 034S 047S 048S 048S 048S T49S A50S S37S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 043S 043S 043S 043S 043S 043S 043	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 97 98 L_not100 99 100 101 L_count10s	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 G _'1' O _'0' Z	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto next iteration if _num is negative prepare for printing numbers reset Acc reset digit load counter subtract 10 goto next iteration if _num is negative piump if not 100 yet goto next iteration if _num is negative piump if not 100 yet goto next iteration if _num is negative piump if not 100 yet goto next iteration if _num is negative goto next iteration iteration iterative goto next iter</pre>
E75S A39S S41S E86S T51S 044S 044S 048S 048S 749S A51S 653S 033S 749S 537S 698S 042S 043S 043S 2S T49S T52S X50S S38S 6109S	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 97 98 L_not100 99 97 98 100 101 102	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_not100 C _'0' Z	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 ctace mumber</pre>
E75S A39S S41S E86S T51S 034S 047S 048S 047S 048S 048S T49S S37S 698S 042S 043S ZS T49S T52S A50S S38S 6109S T51S	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 97 98 L_not100 99 80 90 91 92 93 94 93 94 95 96 96 97	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 G L_not100 O _'1' O _'0' Z _'0' Z0' Tdummy T _d A _cnt S _100 G L_not100 O _'1' O _'0' Z0' Tdummy T _d A _cnt S00 G L_not100 O _'1' O _'0' Z0' C0' Tdummy Td A _cnt S00 G L_not100 O'1' O _'0' Z0	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number</pre>
E75S A39S S41S E86S T51S 034S 047S 048S 048S 048S 048S 048S 048S 749S 533S 033S T49S 537S 698S 042S 043S 042S 043S 042S 043S 043S 043S 043S 043S 043S 043S 043	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 93 94 95 97 97 98 L_not100 99 97 90 88 L_count10s 102 103 104	<pre>E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' C _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _d </pre>	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit impresed digit</pre>
E75S A39S E86S T51S 034S 047S 048S 048S 048S 048S T49S A50S 033S T49S A50S 042S 043S 043S 043S 043S 043S 043S 043S 043	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 97 98 L_not100 99 100 101 1_ccount10s 102 103	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_print10S T _num A _cnt S _10 G L_print10S T _num A _cd A _1	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit </pre>
E75S A39S S41S E86S T51S 044S 044S 048S 048S 749S A51S 653S 033S 749S 537S 698S 042S 043S 042S 043S 043S 2S T49S T52S A50S S38S 6109S T51S S452S A41S	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 97 98 L_count10s 102 103 104 105 106 107	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _Z T _dummy A _cnt S _100 G L_not100 O _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _1 T _d A _1 T _d	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output 8 output 0 output 2 reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit</pre>
E75S A39S S41S E86S T51S 034S 047S 048S 047S 048S 048S T49S S37S 698S 042S 043S 25 T49S T52S A50S S38S 6109S T51S A52S A41S T52S A41S	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 97 98 L_not100 99 97 88 96 97 97 98 L_count105 100 101 L_count105 103	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 G L_not100 O _ '1' O _'0' Z T _dummy T _d A _cnt S _100 G L_print10S T _num A _d A _1 T _d A _num	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number</pre>
E75S A39S S41S E86S T51S 034S 047S 048S 048S 048S 048S 048S 048S 749S A51S 653S 033S T49S 337S 698S 042S 043S 042S 043S 042S 043S 042S 043S 043S 042S 043S 042S 043S 042S 043S 042S 043S 043S 042S 043S 042S 043S 042S 043S 043S 043S 045 045 048S 048S 048S 048S 048S 048S 048S 048S	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 93 94 95 96 97 97 98 L_not100 99 97 88 80 80 80 90 91 92 93 94 95 93 94 95 96 97 97 96 97 97 98 80 80 80 80 80 80 80 80 80 80 80 80 80	<pre>E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' O _'0' Z _'0' T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _1 T _d A _num E L_count10s</pre>	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit load number loop unconditionally</pre>
E75S A39S S41S E86S T51S 034S 047S 048S 048S 048S 048S T49S A50S G53S 042S 043S 043S 043S 043S 042S 043S 042S 043S 043S 043S 043S 043S 043S 043S 043	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 97 98 L_not100 99 100 101 L_count10s 102 103 104 105 106 107 108 109 L_print10s	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 G L_print10S T _num A _d A _1 T _d A _num E L_count10S T _dummy	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number loap unconditionally reset Acc </pre>
E75S A39S S41S E86S T51S 034S 047S 048S 047S 048S 048S T49S 337S 653S 033S T49S 537S 698S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 042S 043S 042S 042S 042S 042S 042S 042S 042S 042	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 97 98 L_not100 99 97 88 80 90 91 92 93 94 95 96 97 97 98 L_not100 99 97 98 L_not100 99 90 100 101 L_count10s 105 106 107 108 109 L_print10s	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _Z T _dummy A _cnt S _100 G L_not100 O _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _1 T _d A _num E L_count10s T _dummy A _d A _1 T _d	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load digit </pre>
E75S A39S S41S E86S T51S 034S 047S 048S 047S 048S 048S T49S 537S 698S 042S 043S 25 749S T52S A50S 537S 5043S 25 749S T52S 538S 6109S T51S A52S A41S T52S A41S T52S A51S E101S T49S	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 97 98 L_not100 99 80 101 L_count10s 102 103 104 105 106 107 106 107 108 109 L_print10s	<pre>E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _1 T _d A _num E L_count10s T _dummy A _d S _1</pre>	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load number load digit decrease digit by 1 </pre>
E75S A39S S41S E86S T51S 034S 047S 048S 048S 048S 048S T49S A50S 033S T49S A50S 033S T49S A50S 042S 042S 042S 042S 042S 042S 043S 2S T49S T52S A50S S38S G109S T51S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S A51S A51S A51S A52S A51S A51S A52S A53S A53S A53S A53S A53S A53S A53S A53	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 93 94 95 96 97 97 98 L_not100 99 100 101 L_count10s 102 103 104 105 106 107 108 109 101 111	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z _ T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _num E L_count10s T _dummy A _d S _1 G L_1 G L_1 G L_1	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number loop unconditionally reset Acc load digit if negative (digit was 0), skip printing of tens digits</pre>
E75S A39S S41S E86S T51S 044S 044S 048S 048S 048S T49S A51S 653S 033S T49S A50S S37S 698S 043S 043S 043S 043S 043S 043S C43S 043S C43S 043S C53S 043S C43S C43S C43S C43S C53S C43S C43S C43S C43S C43S C43S C43S C4	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 97 98 L_not100 99 100 101 L_count10s 102 103 104 105 106 107 108 109 L_print10s	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_print10S T _num A _d A _1 T _d A _1 T _dummy A _1 A _1 G L_1 A _1 A _1 A _1 A _1 A _1	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print los if Acc < 0 store number load digit increase digit store digit load number load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1</pre>
E75S A39S S41S E86S T51S 034S 047S 048S 047S 048S 048S T49S 337S 698S 042S 043S 042S 042S 042S 042S 042S 042S 042S 042	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 97 98 L_count10s 100 101 L_count10s 105 106 107 108 109 L_print10s 104 111 112 113 114	E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z _ T _dummy T _d A _cnt S _100 G L_not100 O _'1' O _'0' Z _ T _dummy T _d A _cnt S _10 G L_print10S T _num A _d A _1 T _d A _num E L_count10S T _dummy A _d S _1 G L_1 A _1 L _2^(11-2)	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure</pre>
E75S A39S S41S E86S T51S 034S 047S 048S 047S 048S 048S T49S 537S 698S 042S 043S 25 749S T52S A50S 537S 698S 042S 043S 25 T49S T52S A50S 538S 6109S T51S A52S A41S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T55S A51S E101S T55S A51S A51S A51S A51S A51S A51S A51S A	76 77 78 79 80 81 82 83 84 85 86 85 86 87 87 88 89 89 90 91 92 93 94 95 96 97 98 88 90 91 92 93 94 95 96 97 98 88 90 91 92 93 94 95 96 97 98 96 97 98 90 97 98 90 90 91 92 93 94 95 96 97 96 97 98 1_not100 99 100 101 1_ccount10s 106 107 106 107 108 109 109 100 101 111 112 113 114 115	<pre>E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' C _'0' C _'0' T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _1 T _d A _num E L_count10s T _dummy A _d S _1 G L_1 A _1 L 2^(11-2) T _d</pre>	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc load counter subtract 10 goto print los if Acc < 0 store number load digit increase digit store digit load number load digit increase digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save printable figure subtract 0 if acc is the store if a store is the store is the store is the store is the store of the store is the store of the store is printing of tens digits rest Acc load digit if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure subtract is a figure subtract is a store is a</pre>
E75S A39S S41S E86S T51S 034S 047S 048S 048S 048S T49S A50S 033S T49S A50S 033S T49S A50S 033S T49S A50S C33S 042S 042S 042S 042S 043S 2S T49S T52S A50S S38S G109S T51S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T52S A51S E101S T52S A51S E101S T52S A51S E101S T52S A51S E101S T52S A51S E101S T52S A51S E101S T52S A51S E101S T52S A51S E101S T52S A51S C52S A51S C52S A51S C52S A51S C52S A51S C52S A51S C52S A51S C52S A51S C52S A51S C53S C53S C53S C53S C53S C53S C53S C53	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 100 L_count10s 102 103 104 105 106 107 108 109 L_printl0s 106 107 108 109 111 112 113 114 115 116	<pre>E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' O _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _1 T _d A _num E L_count10s T _dummy A _d S _1 G L_1 A _1 L 2^(11-2) T _d O _d </pre>	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc load counter subtract 10 goto print los if Acc < 0 store number load digit store digit load number load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save printable figure print figure / digit </pre>
E75S A39S S41S E86S T51S 044S 047S 048S 048S 048S T49S A51S 653S 033S T49S A50S S37S 698S 043S 043S 043S 043S 043S 043S 043S 043	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 98 100 101 L_count10s 102 103 104 105 106 107 108 109 L_printlos 106 107 108 109 L_printlos 106 107 108 109 L_printlos 106 107 108 109 L_printlos 106 107 108 109 L_printlos 106 107 108 109 L_printlos 106 107 108 109 L_printlos 106 107 108 109 L_printlos 106 107 108 109 L_printlos 106 107 108 109 L_printlos 106 107 108 109 L_printlos 106 107 108 109 L_printlos 106 107 108 109 107 108 109 107 108 107 108 109 107 108 107 107 108 107 107 107 107 108 108 108 108 108 108 108 108	<pre>E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _1 T _d A _num E L_count10s T _dummy A _d S _1 G L_1 A _1 L 2^(11-2) T _d U C _0 T _dummy</pre>	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 output 0 end the program reset Acc load counter subtract 10 goto print 10s if Acc < 0 store number load digit store digit load number load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure print figure / digit reset Acc </pre>
E75S A39S S41S E86S T51S 034S 047S 048S 047S 048S 048S T49S 337S 653S 033S T49S 537S 698S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 043S 042S 042S 042S 042S 042S 042S 042S 042	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 97 98 L_count10s 100 101 L_count10s 102 103 104 105 106 107 108 109 L_printlus 109 111 112 113 114 115 116 117 L_1: 118	<pre>E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _l T _d A _num E L_count10s T _dummy A _d S _1 G L_1 A _1 L 2^(11-2) T _d N _d N A _num </pre>	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load digit increase digit by 1 if negative (digit was 0), skip printing of tens digits rest Acc load digit reset Acc load digit reset Acc load number loap unconditionally reset Acc load number subtract 10 figure printable figure print figure / digit reset Acc load number </pre>
E75S A39S S41S E86S T51S 034S 047S 048S 047S 048S 048S T49S 537S 033S T49S 537S 043S 25 043S 25 749S T52S 043S 25 749S T52S A50S 538S 6109S T51S A52S A41S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T49S T52S A51S E101S T52S A51S E101S T52S A51S E101S T52S A51S E101S T52S A51S E101S T53S A51S E101S T53S A51S E101S T53S A51S E101S T53S A51S E101S T53S A51S C53S C53S C53S C53S C53S C53S C53S C53	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 97 98 L_not100 99 100 101 L_count10s 102 103 104 105 106 107 108 109 L_print10s 106 107 108 109 L_print10s 104 111 112 113 114 115 116 117 L_1: 118 119	<pre>E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' C _'0' C _'0' T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _1 T _d A _num E L_count10s T _dummy A _d S _1 G L_1 A _1 L 2^(11-2) T _d N _num L 2^((11-2)</pre>	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 1 output 0 output 0 end the program reset Acc load counter subtract 10 goto print 10s if Acc < 0 store number load digit load number load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits reset Acc load digit decrease digit, by increasing with 1 Acc << 11, create a printable figure </pre>
E75S A39S S41S E86S T51S 034S 047S 048S 048S 048S 048S T49S A51S 653S 033S T49S A50S 698S 042S 042S 042S 042S 042S 042S 043S 25 T49S T52S A50S 538S 6109S T51S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S E101S T49S A52S A51S C52S A51S C52S A51S C52S A51S C52S A51S C52S A51S C52S A51S C53S C53S C53S C53S C53S C53S C53S C53	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 97 98 L_not100 99 100 101 L_count10s 102 103 104 105 106 107 108 109 111 112 113 114 115 116 117 L_1: 118 119 120	<pre>E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' O _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _1 T _d A _num E L_count10s T _dummy A _d S _1 G L_1 A _1 L 2^(11-2) T _d U S _num L 2^(11-2) T _d</pre>	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc load digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load digit if negative (digit was 0), skip printing of tens digits resst Acc load digit if negative (digit must 0, skip printing of tens digits resset Acc load number Acc < 11, create a printable figure save printable figure Acc < 11, create a printable figure save printable figure save printable figure </pre>
E75S A39S S41S E86S T51S 044S 047S 048S 048S T49S A51S 653S 033S T49S A50S S37S 698S 043S 043S 043S 043S 043S 043S 043S 043	76 77 78 79 80 81 82 83 84 85 86 L_notBuzz 87 88 89 L_printNum 90 91 92 93 94 95 96 97 98 L_not100 99 98 100 101 L_count10s 102 103 104 105 106 107 108 109 L_printlos 106 107 115 116 117 L_1: 118 119 120 121	<pre>E L_Buzz A _5 S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 G L_print10s T _num A _d A _1 T _d A _num E L_count10s T _dummy A _d S _1 G L_1 A _1 L 2^(11-2) T _d N _num L 2^(11-2) T _d O _d </pre>	<pre>loop until Acc < 0 add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits reset Acc load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits reset Acc load number Acc <: 11, create a printable figure save printable figure print figure / digit </pre>

XS 123 L_end

Х

"FizzBuzz" on the EDSAC / Initial Orders I

T49S	73 L_notFizz	T _dummy	r
A50S	74	A _cnt	l
S39S	75 L_Buzz	S _5	S
E75S	76	E L_Buzz	l
A39S	77	A _5	а
S41S	78	S _1	S
E86S	79	E L_notBuzz	j
T51S	80	T _num	s
034S	81	0 _LS	р
044S	82	0 _B	0
047S	83	0 _U	0
048S	84	0 _Z	0
048S	85	0 _Z	0
T49S	86 L_notBuzz	T _dummy	r
A51S	87	A _num	l
G53S	88	G L_next	g
033S	89 L_printNum	0 _FS	р
T49S	90	T _dummy	r
A50S	91	A _cnt	l
S37S	92	S _100	S
G98S	93	G L_not100	j
042S	94	0 _'1'	0
043S	95	0 _'0'	0
043S	96	0 _'0'	0
ZS	97	Z	e

- . . .

eset Acc load Acc with _cnt subtract 5 loop until Acc < 0add 5, restore previous value subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U butput Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0

end the program

T100C				
11235	31		I L_end	mark end of program
E60S	32		E L_start	jump to the beginning of program
#S	33	_FS	#	figure shift
*S	34	_LS	*	letter shift
&S	35	LF	&	linefeed character
as	36	CR	a	carriage return character
P1005	37	100	P 100	constant 100
P105	20	10	P 10	constant 10
1105	20	_10	n 10	
P55	39	_5	P 5	constant 5
P3S	40	_3	P 3	constant 3
P1S	41	_1	P 1	constant 1
QS	42	_'1'	Q	constant figure 1
PS	43	_'0'	Р	constant figure 0
BS	44	_B	В	constant letter B
FS	45	F	F	constant letter F
IS	46	ī	I	constant letter I
US	47	U	U.	constant letter U
75	18	_0	7	constant letter 7
23	40		2	used to fluch and react the accumulator
FJ D1C	49	_uummy		asunter surrent number to be considered will be increased
F13	50			counter, current number to be considered, write be increased
P3	51	_num	r D	number to be printed, negative in counter is mod 3 or mod 5
PS	52	_a	P	
0345	53	L_next	0_LS	output LS, prepare for printing letters
0355	54		0 _LF	output LF, linefeed
0365	55		0 _CR	output CR, carriage return
T49S	56		T _dummy	reset Acc
A50S	57		A _cnt	load Acc with _cnt
A41S	58		A _1	increase Acc
T50S	59		T _cnt	store Acc into _cnt, reset Acc
A50S	60	L_start	A _cnt	load Acc with _cnt (we know that Acc initially is 0)
U51S	61		U _num	tentatively set number to be printed
\$40\$	62	L_tryFizz	S _3	subtract 3
E62S	63	- ,	– E L trvFizz	loop until Acc < 0
A40S	64		A 3	add 3. restore previous value
5415	65		S 1	subtract 1 to check if $Acc was 0$
E725	60		5 _ 1 E potEizz	$\frac{1}{1000}$ if Acc was not 0, in number was not divisable by 2
L/33 TE10	67			sot num to possible value flag that no value should be printed
1515	67			set _num to negative value, itag that no value should be printed
0345	68		0 _LS	prepare printing letters
0455	69		0_F	output F
0465	70		0_1	output 1
0485	71		0 _Z	output Z
048S	72		0 _Z	output Z
T49S	73	L_notFizz	T _dummy	reset Acc
A50S	74		A _cnt	load Acc with _cnt
S39S	75	L_Buzz	S _5	subtract 5
E75S	76		E L_Buzz	loop until Acc < 0
A395	77		A _5	add 5, restore previous value
1.000				
S41S	78		S _1	SUDTRACT 1, TO CHECK IT ACC WAS 0
S41S E86S	78 79		S _1 E L_notBuzz	jump if Acc was not 0, ie number was not divisable by 5
S41S E86S T51S	78 79 80		S _1 E L_notBuzz T num	subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set num to negative value. flag that no value should be printed
S41S E86S T51S 034S	78 79 80 81		S _1 E L_notBuzz T _num O LS	subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters
S41S E86S T51S 034S 044S	78 79 80 81 82		S _1 E L_notBuzz T _num O _LS O _B	Subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B
S41S E86S T51S 034S 044S 047S	78 79 80 81 82 83		S _1 E L_notBuzz T _num O _LS O _B O _U	subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U
S41S E86S T51S 034S 044S 047S	78 79 80 81 82 83		S _1 E L_notBuzz T _num O _LS O _B O _U O _Z	subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output 7
S41S E86S T51S 034S 044S 047S 048S	78 79 80 81 82 83 83		S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _7	subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z
S41S E86S T51S 034S 044S 047S 048S 048S	78 79 80 81 82 83 84 85	L notRung	S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z C _Z	Subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z
S41S E86S T51S 034S 044S 044S 044S 048S T49S	78 79 80 81 82 83 84 85 86	L_notBuzz	S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy	Subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc
S41S E86S T51S 034S 044S 044S 044S 048S T49S A51S	78 79 80 81 82 83 84 85 86 87	L_notBuzz	S _1 E L_norBuzz T_num O _LS O _B O _U O _Z O _Z T_dummy A _num	subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed
S41S E86S T51S 034S 044S 044S 044S 048S 048S T49S A51S G53S	78 79 80 81 82 83 84 85 86 87 88	L_notBuzz	S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next	subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative
S41S E86S T51S 034S 044S 047S 048S 048S 048S T49S A51S G53S 033S	78 79 80 81 82 83 84 85 86 87 88 89	L_notBuzz L_printNum	S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS	subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers
S41S E86S T51S 034S 044S 044S 048S 048S T49S A51S G53S 033S T49S	78 79 80 81 82 83 84 85 86 87 88 89 90	L_notBuzz L_printNum	S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy	Subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc
S41S E86S T51S 034S 044S 047S 048S 048S T49S A51S 033S T49S A50S	78 79 80 81 82 83 84 85 86 87 88 89 90 91	L_notBuzz L_printNum	S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt	subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter
 S415 S415 E865 T515 O345 O445 O485 O485 O485 O485 T495 A505 S375 	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92	L_notBuzz L_printNum	S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100	subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop
S415 E86S T515 O445 O44S O48S O48S T49S A515 G53S O33S T49S A50S S37S G98S	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93	L_notBuzz L_printNum	S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet</pre>
S415 E86S T51S 044S 044S 047S 048S T49S A51S 653S 033S T49S A50S S37S 698S 042S	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94	L_notBuzz L_printNum	S _1 E L_nontBuzz T_num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1'	Subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1
 S415 S415 E86S T515 O445 O445 O475 O485 O485 O485 O485 O495 A505 S375 G985 O425 O435 	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94	L_notBuzz L_printNum	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0'</pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0</pre>
S415 E86S T515 O445 O445 O445 O445 O445 O485 A515 G535 O335 T495 A505 S375 G985 O425 O435 O435	78 79 80 81 82 83 84 85 86 87 88 87 88 89 90 91 92 93 94 95 96	L_notBuzz L_printNum	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z 0 _Z 7 _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0'</pre>	Subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0
 S415 S415 E86S T515 O445 O445 O4485 O485 T495 A515 G535 O335 T495 A505 S375 G985 O425 O435 O435 Z5 	78 79 80 81 82 83 84 85 86 87 88 87 88 89 90 91 92 93 94 95 96 97	L_notBuzz L_printNum	S _1 E L_notBuzz T_num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T_dummy A _num G L_next 0 _FS T_dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' Z	Subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program
S415 S415 E86S T515 044S 044S 044S 048S T49S 653S 033S T49S S37S 698S 042S 043S 043S 043S 043S 2S T49S	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98	L_notBuzz L_printNum	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' Z T _dummy T _dummy</pre>	Subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc
S415 S415 E86S T515 0345 0445 0475 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0435 0435 0435 25 T495	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 92 93 945 96 97 98	L_notBuzz L_printNum L_not100	S _1 E L_notBuzz T _num O _LS O _B O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d	Subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 end the program reset Acc
S415 S415 E86S T515 0345 0445 0445 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0335 7495 7495 7505	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 97 98 99 9100	L_notBuzz L_printNum L_not100	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 7 _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' Z T _dummy T _d A _cnt</pre>	Subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc load counter
S415 S415 E86S T515 O44S O44S O44S O48S T49S S37S G98S O42S O43S ZS T49S T52S S38S	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 9100 101	L_notBuzz L_printNum L_not100 L_count10s	S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' Z T _dummy T _d A _cnt S _10	Subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10
S415 S415 E86S T515 O44S O44S O44S O48S T49S S37S G33S T49S S37S G98S O42S O43S O43S O43S O43S O43S C53S G98S O42S O43S C53S G98S O42S O43S O43S O43S O43S O43S O43S O43S O43	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101	L_notBuzz L_printNum L_not100 L_count10s	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' 2 T _dummy T _d A _cnt S _10 G A _cnt </pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0</pre>
S415 S415 E86S T515 0345 0445 0445 0445 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0505 3385 0515	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101	L_notBuzz L_printNum L_not100 L_count10s	<pre>S _1 E L_notBuzz T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 T _dummy T _dumm</pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc load counter subtract 10 goto print 10s if Acc < 0 store number</pre>
S415 S415 E86S T515 0345 0445 0445 0445 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0335 7495 7525 S385 61095 7515 5385 61095 7525	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 97 98 99 100 101 102	L_notBuzz L_printNum L_not100 L_count10s	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 7 _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' Z T _dummy T _d A _cnt S _10 G L_not1 G L_not1 G _Lnot1 G _Lnot1 A _cnt A _c</pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 1 output 0 output 1 load counter subtract 10 goto print 10s if Acc < 0 store number load divit</pre>
S415 S415 E86S T515 O44S O44S O44S O44S O48S T49S S37S G98S O42S O43S ZS T49S T52S S38S G109S T51S S38S G109S T51S A52S	78 79 80 81 82 83 84 85 86 87 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104	L_notBuzz L_printNum L_not100 L_count10s	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _1</pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit</pre>
S415 S415 E86S T515 0345 0445 0485 0425 0425 0425 0435 0435 0435 0435 0435 0435 0435 0425 0425 0425 0425 0425 0425 0425 0425 0425 0425 0425 0425	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 97 910 101 102 103 104	L_notBuzz L_printNum L_not100 L_count10s	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' 0 _'0' Z T _dummy T _d A _cnt S _10 G L_printl0s T _num A _d A _d A _d A _d A _d </pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit</pre>
S415 S415 E86S T515 O445 O445 O445 O445 O445 C485 A515 G535 O485 C485 C485 C485 C485 C485 C485 C435 C435 C435 C435 C435 C435 C435 C43	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104	L_notBuzz L_printNum L_not100 L_count10s	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' Z T _dummy T _d A _cnt S _10 G L_printl0s T _num A _d A _1 T _d A _num</pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number</pre>
S415 S415 E86S T515 0345 0445 0445 0445 0445 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0495 A505 S375 0985 0425 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 3585 51095 3885 61095 7515 4525 4415 7526	78 79 800 81 82 83 84 85 86 87 88 87 92 93 94 92 93 94 95 96 97 97 98 97 91 102 103 104 105 106 107	L_notBuzz L_printNum L_not100 L_count10s	<pre>S _1 E L_notBuzz T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _cnt S _10 G L_print10s T _num A _d A _1 T _d A _num E L _count10</pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number</pre>
S415 S415 E86S T515 O44S O44S O44S O44S O48S T49S S37S G98S O42S O43S C42S O43S C42S O43S C42S O43S C43S C43S C43S C43S C43S C43S C43S C	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 90 101 102 103 104 106 107	L_notBuzz L_printNum L_not100 L_count10s	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' Z T _dummy T _d A _cnt S _10 G L_printl0s T _num A _d A _1 T _d A _num E L_countl0s</pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number loog unconditionally </pre>
S415 S415 E86S T515 O445 O445 O445 O445 O485 O385 A505 S375 G985 O425 O438 C5 A505 S375 G985 C425 O435 O435 C425 O435 C5 S4505 S385 G1095 T515 A525 A415 T525 A515 E1015 T495	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 87 99 90 90 97 90 97 90 90 90 90 90 90 90 90 90 90 90 90 90	L_notBuzz L_printNum L_not100 L_count10s	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' 0 _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _1 T _d A _num E L_count10s T _dummy</pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number loop unconditionally reset Acc load die the program load store load digit increase Acc load digit load number loop unconditionally reset Acc load counter load digit </pre>
S415 S415 E86S T515 0345 0445 0445 0485 0435	78 79 80 81 82 83 84 85 86 87 88 89 91 92 93 94 95 96 97 96 97 90 100 101 102 103 104 105 106 107 107 80 90 91 90 90 91 90 90 90 90 90 90 90 90 90 90 90 90 90	L_notBuzz L_printNum L_not100 L_count10s	<pre>S _1 E L_notBuzz T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _1 T _d A _1 T _d A _1 C L_print10s T _num E L_count10s T _dummy A _1 C L_print10s T _num E L_count10s T _dummy A _1 C L_count10s T _dummy A _1 C L_count10s T _dummy A _1 C L_count10s C L_count10s C _1 C _1 C _1 C _1 C _1 C _1 C _1 C _1</pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load digit increase digit load number load digit increase digit load number load digit increase Acc load digit </pre>
S415 S415 E86S T515 0345 0445 0445 0485 0435 25 1495 1525 A515 E1015 1495 1525 A525 5415	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 97 96 97 97 98 99 910 101 102 103 104 105 106 107 108 109 100 101 105 109 100 100 100 100 100 100 100 100 100	L_notBuzz L_printNum L_not100 L_count10s L_print10s	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' Z T _dummy T _d A _cnt S _10 G L_printl0s T _num A _1 T _d A _1 T _d A _num E L_count10s T _dummy A _d S _1</pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load number load digit decrease digit by 1</pre>
S415 S415 E86S T515 0345 0445 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0335 T495 525 A515 61015 7495 5415 61175	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 97 98 99 100 101 102 103 104 105 106 107 108 109 111 112	L_notBuzz L_printNum L_not100 L_count10s	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' Z T</pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number loop unconditionally reset Acc load digit if negative (digit was 0), skip printing of tens digits</pre>
S415 S415 E86S T515 O445 O445 O445 O445 O485 O485 O485 O485 O485 O485 O485 O485 O485 O385 O495 O435 O435 <t< td=""><td>78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 96 97 96 97 9100 101 102 103 104 105 106 107 108 109 110 1112 113</td><td>L_notBuzz L_printNum L_not100 L_count10s</td><td><pre>S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' O _'0' Z T _dummy T _d A _cnt S _10 G L_print108 T _num A _d A _num E L_count108 T _dummy A _d A _num E L_count108 T _dummy A _d S _1 G L_1 A _1</pre></td><td><pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load digit increase digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1</pre></td></t<>	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 96 97 96 97 9100 101 102 103 104 105 106 107 108 109 110 1112 113	L_notBuzz L_printNum L_not100 L_count10s	<pre>S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' O _'0' Z T _dummy T _d A _cnt S _10 G L_print108 T _num A _d A _num E L_count108 T _dummy A _d A _num E L_count108 T _dummy A _d S _1 G L_1 A _1</pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load digit increase digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1</pre>
S415 S415 E86S T515 0345 0445 0445 0485 A4515 6535 0335 T495 A515 6985 0425 0435 25 T495 A525 A415 T525 A515 E1015 A525 S415 E125	78 79 800 81 82 83 84 85 86 87 88 89 91 92 93 94 95 96 97 95 96 97 9100 101 102 103 104 105 106 107 110 111 112 113 114	L_notBuzz L_printNum L_not100 L_count10s L_print10s	<pre>S _1 E L_notBuzz T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num E L_count10s T _dummy A _d S _1 C _1 T _dummy A _d A _1 T _1 T _1 T _1 T _1 T _1 T _1 T _1 T</pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure</pre>
S415 S415 E86S T515 O44S O44S O44S O44S O48S T49S S37S G98S O42S O43S Z5 T49S T52S A50S S38S G109S T51S A50S S38S G109S T51S A50S S38S G109S T51S A51S E101S T49S S41S E101S T49S S41S E101S T49S S41S S41S S41S S41S S41S S41S S41S S	78 79 800 81 82 83 84 85 86 87 88 89 90 91 92 93 94 97 96 97 97 98 99 90 101 102 103 104 105 106 107 108 109 110 111 112 113	L_notBuzz L_printNum L_not100 L_count10s L_print10s	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _1 T _d A _num E L_count10s T _dummy A _d S _1 C _1 T _dummy A _1 T _d A _num E L_count10s T _dumy A _1 L _2 (11-2) T _d L _2 (11-2) T _d</pre>	<pre>subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save printable</pre>
S415 S415 E86S T515 0345 0445 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0335 T495 525 0425 0425 0425 0425 0425 0425 0435 225 0435 255 0425 0435 25 0435 255 0525	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113	L_notBuzz L_printNum L_not100 L_count10s	<pre>S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 G L_print10s T _num A _d A _num E L_count10s T _dummy A _d S _1 G L_print10s T _dummy A _d S _1 G L_print10s T _dummy A _d A _num E L_count10s T _dummy A _d S _1 G L_1 A _1 L 2^(11-2) T _d A _d C _1 C _1 C _1 C _1 C _1 C _1 C _1 C _1</pre>	<pre>Subtract 1, to check if Acc Was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load digit increase digit by 1 if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure print figure / digit</pre>
S415 S415 S415 E86S T515 0345 0445 0445 0445 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 5105 3415 1415 15125 0425 1495 1495	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 96 97 98 99 91 100 101 102 103 104 105 106 107 110 111 111 111 111 111 111	L_notBuzz L_printNum L_not100 L_count10s L_print10s	<pre>S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' O _'0' Z T _dummy T _d A _cnt S _10 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_not100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_not100 S _10 G L_not100 G L</pre>	<pre>Subtract 1, to check if Acc Was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output Z output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print los if Acc < 0 store number load digit increase digit store digit load number load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits resore digit, by increasing with 1 Acc << 11, create a printable figure prest Acc</pre>
S415 S415 S415 E86S T515 0345 0445 0445 0485 0435 0415 0415	78 79 800 81 82 83 84 85 86 87 88 87 99 91 92 93 94 95 96 97 97 98 99 9100 101 102 103 104 105 106 107 110 111 112 113 114 115 116 117 118	L_notBuzz L_printNum L_not100 L_count10s L_print10s	<pre>S _1 E L_notBuzz T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num E L_count10s T _num E L_count10s T _dummy A _d S _1 G L_1 A _1 L 2^(11-2) T _d A _1 L 2^(11-2) T _dummy A _num</pre>	<pre>Subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits rester Acc load ingit if negative (digit reset Acc load number load ingit if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure save printable figure print figure / digit reset Acc load number</pre>
S415 S415 S415 E865 T515 0345 0445 0445 0485 0425 0425 0425 0425 0425 0425 0425 0425 0425 0425 0425 0425 0425 0425 0425 0425 0425 0425 0425 0415 1525 0525	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 111 112 113 114 115 116 117 7 118	L_notBuzz L_printNum L_not100 L_count10s L_print10s L_1:	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dummy A _num G L_next 0 _FS T _dummy A _cnt S _100 G L_not100 0 _'1' 0 _'0' Z T _dummy T _d A _cnt S _10 G L_print10s T _num A _d A _num E L_count10s T _dummy A _d S _1 C L_1 A _num E L_count10s T _dummy A _d S _1 C L_1 C L_</pre>	<pre>Subtract 1, to check 1f Acc Was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 1 output 0 output 0 end the program reset Acc reset digit load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number loop unconditionally reset Acc load digit if negative (digit was 0), skip printing of tens digits restore digit, by increasing with 1 Acc << 11, create a printable figure print figure / digit reset Acc load number Acc << 11, create a printable figure</pre>
S415 S415 S415 E86S T515 0345 0445 0475 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0485 0495 1495 1525 0438 0515	78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120	L_notBuzz L_printNum L_not100 L_count10s L_print10s	<pre>S _1 E L_notBuzz T _num O _LS O _B O _U O _Z O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _100 G L_print108 T _num A _d A _cnt S _10 G L_print108 T _num A _d A _num E L_count108 T _dummy A _d S _1 G L_1 A _num E L_count108 T _dummy A _d S _1 G L_1 A _1 L 2^(11-2) T _d C _2(11-2) T _d C _2(11-2) T _d</pre>	<pre>Subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 goto print 10s if Acc < 0 store number load digit increase digit by 1 if negative (digit was 0), skip printing of tens digits rester Acc load digit if negative (digit was 0), skip printing of tens digits rester digit load number Acc << 11, create a printable figure save printable</pre>
S415 S415 S415 E86S T515 0345 0445 0445 0485 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0415 15125 0525 0415 1495 04175	78 79 80 81 82 83 84 85 86 87 88 87 99 91 92 93 94 95 96 97 98 99 91 100 101 102 103 104 105 106 107 110 111 112 113 114 115 116 117 118 119 120	L_notBuzz L_printNum L_not100 L_count10s L_print10s	<pre>S _1 E L_notBuzz T _num O _LS O _B O _U O _Z T _dummy A _num G L_next O _FS T _dummy A _cnt S _100 G L_not100 O _'1' O _'0' O _'0' Z T _dummy T _d A _cnt S _10 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_not100 O _'1' O _'0' Z T _dummy T _d A _cnt S _10 G L_not100 O _'1' O _'0' Z Z Z Z Z Z Z Z Z Z Z Z Z Z Z Z Z Z</pre>	<pre>Subtract 1, to check if Acc was 0 jump if Acc was not 0, is number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 output 0 end the program reset Acc load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number load digit if negative (digit was 0), skip printing of tens digits reset Acc load digit if negative (digit was 0), skip printing of tens digits reset Acc load number Acc < 11, create a printable figure save printable figure print figure / digit </pre>
S415 S415 S415 E865 T515 0345 0445 0485 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0435 0525 0415 1525 0415 1525 0415 1525 0525 1525	78 79 800 81 82 83 84 85 86 87 88 87 88 89 90 91 92 93 94 95 96 97 95 96 97 91 00 101 102 103 104 105 106 107 110 111 112 120 122	L_notBuzz L_printNum L_not100 L_count10s L_print10s	<pre>S _1 E L_notBuzz T _num 0 _LS 0 _B 0 _U 0 _Z 0 _Z T _dunmy A _num G L_next 0 _FS T _dunmy A _cnt S _100 G L_not100 0 _'1' 0 _'0' 2 T _dunmy T _d A _cnt S _100 G L_not100 0 _'1' 0 _'0' 2 T _dunmy T _d A _cnt S _10 G L_print10S T _num A _d A _1 T _d A _num E L_count10S T _dunmy A _d S _1 G L_print10S T _dunmy A _d A _1 T _d A _1 T _d A _1 Z T _d C _1 Z T _dumy A _d A _1 T _d A _1</pre>	<pre>Subtract 1, to check if Acc was 0 jump if Acc was not 0, ie number was not divisable by 5 set _num to negative value, flag that no value should be printed prepare printing letters output B output U output Z reset Acc load _num to check number to be printed goto next iteration if _num is negative prepare for printing numbers reset Acc load counter subtract 100, check if we should stop jump if not 100 yet output 0 output 0 end the program reset Acc load counter subtract 10 goto print 10s if Acc < 0 store number load digit increase digit store digit load number loop unconditionally reset Acc load digit decrease digit by 1 if negative (digit was 0), skip printing of tens digits restore digit reset Acc load number Acc << 11, create a printable figure save printable figure print figure / digit unconditionally interest acc load number Acc << 11, create a printable figure save printable figure print figure / digit unconditionally interest Acc load number Acc << 11, create a printable figure save printable figure print figure / digit unconditionally interest Acc load number Acc << 11, create a printable figure save printable figure print figure / digit unconditionally interest Acc load number Acc << 11, create a printable figure save printable figure save printable figure print figure / digit unconditionally </pre>

T49S	98 L_not	:100 T	_dummy	reset Acc
T52S	99	Т	_d	reset digit
A50S	100	A	_cnt	load counter
S38S	101 L_cou	int10s S	_10	subtract 10
G109S	102	G	L_print10s	goto print 10s
T51S	103	Т	_num	store number
A52S	104	A	_d	load digit
A41S	105	A	_1	increase digit
T52S	106	Т	_d	store digit
A51S	107	A	_num	load number
E101S	108	E	L_count10s	loop unconditio
T49S	109 L_pri	int10s T	_dummy	reset Acc
A52S	110	A	_d	load digit
S41S	111	S	_1	decrease digit
G117S	112	G	L_1	if negative (di
A41S	113	A	_1	restore digit,
L512S	114	L	2^(11-2)	Acc << 11, crea
T52S	115	Т	_d	save printable
052S	116	0	_d	print figure /
T49S	117 L_1:	Т	_dummy	reset Acc
A51S	118	A	_num	load number
L512S	119	L	2^(11-2)	Acc << 11, crea
T52S	120	Т	_d	save printable
052S	121	0	_d	print figure /
E53S	122	E	L_next	unconditional j
XS	123 L_enc	l X		

10s if Acc < 0

ditionally

digit by 1 ve (digit was 0), skip printing of tens digits igit, by increasing with 1 , create a printable figure table figure ure / digit er , create a printable figure table figure ure / digit onal jump

T123SE60S#S*S&S@SP100SP10SP5SP3SP1SQSPSBSFSISU SZSPSP1SPSPS034S035S036ST49SA50SA41ST50SA50SU5 1SS40SE62SA40SS41SE73ST51S034S045S046S048S048S T49SA50SS39SE75SA39SS41SE86ST51S034S044S047S04 8S048ST49SA51SG53S033ST49SA50SS37SG98S042S043S 043SZST49ST52SA50SS38SG109ST51SA52SA41ST52SA51 SE101ST49SA52SS41SG117SA41SL512ST52S052ST49SA5 1SL512ST52S052SE53SXS

T123SE60S#S*S&S@SP100SP10SP5SP3SP1SQSPSBSFSISU SZSPSP1SPSPS034S035S036ST49SA50SA41ST50SA50SU5 1SS40SE62SA40SS41SE73ST51S034S045S046S048S048S T49SA50SS39SE75SA39SS41SE86ST51S034S044S047S04 8S048ST49SA51SG53S033ST49SA50SS37SG98S042S043S 043SZST49ST52SA50SS38SG109ST51SA52SA41ST52SA51 SE101ST49SA52SS41SG117SA41SL512ST52S052ST49SA5 1SL512ST52S052SE53SXS

T123SE60S#S*S&S@SP100SP10SP5SP3SP1SQSPSBSFSISU SZSPSP1SPSPS034S035S036ST49SA50SA41ST50SA50SU5 1SS40SE62SA40SS41SE73ST51S034S045S046S048S048S T49SA50SS39SE75SA39SS41SE86ST51S034S044S047S04 8S048ST49SA51SG53S033ST49SA50SS37SG98S042S043S 043SZST49ST52SA50SS38SG109ST51SA52SA41ST52SA51 SE101ST49SA52SS41SG117SA41SL512ST52S052ST49SA5 1SL512ST52S052SE53SXS

Try this program on NISHIO Hirokazu's EDSAC Simulator http://nhiro.org/learn_language/repos/EDSAC-on-browser/index.html There is a small bug in the program. Did you notice?

T123SE60S#S*S&S@SP100SP10SP5SP3SP1SQSPSBSFSISU SZSPSP1SPSPS034S035S036ST49SA50SA41ST50SA50SU5 1SS40SE62SA40SS41SE73ST51S034S045S046S048S048S T49SA50SS39SE75SA39SS41SE86ST51S034S044S047S04 8S048ST49SA51SG53S033ST49SA50SS37SG98S042S043S 043SZST49ST52SA50SS38SG109ST51SA52SA41ST52SA51 SE101ST49SA52SS41SG117SA41SL512ST52S052ST49SA5 1SL512ST52S052SE53SXS

T123SE60S#S*S&S@SP100SP10SP5SP3SP1SQSPSBSFSISU SZSPSP1SPSPS034S035S036ST49SA50SA41ST50SA50SU5 1SS40SE62SA40SS41SE73ST51S034S045S046S048S048S T49SA50SS39SE75SA39SS41SE86ST51S034S044S047S04 8S048ST49SA51SG53S033ST49SA50SS37SG98S042S043S 043SZST49ST52SA50SS38SG109ST51SA52SA41ST52SA51 SE101ST49SA52SS41SG117SA41SL512ST52S052ST49SA5 1SL512ST52S052SE53SXS Here is a quick and dirty fix!

T123SE60S#S*S&S@SP100SP10SP5SP3SP1SQSPSBSFSISU SZSPSP1SPSPS034S035S036ST49SA50SA41ST50SA50SU5 1SS40SE62SA40SS41SE73ST51S034S045S046S048S048S T49SA50SS39SE75SA39SS41SE86ST51S034S044S047S04 8S048ST49SA51SG53S033ST49SA50SS37SA41SG98SZS04 3S043ST49ST52SA50SS38SG109ST51SA52SA41ST52SA51 SE101ST49SA52SS41SG117SA41SL512ST52S052ST49SA5 1SL512ST52S052SE53SXS

T123SE60S#S*S&S@SP100SP10SP5SP3SP1SQSPSBSFSISU SZSPSP1SPSPS034S035S036ST49SA50SA41ST50SA50SU5 1SS40SE62SA40SS41SE73ST51S034S045S046S048S048S T49SA50SS39SE75SA39SS41SE86ST51S034S044S047S04 8S048ST49SA51SG53S033ST49SA50SS37SA41SG98SZS04 3S043ST49ST52SA50SS38SG109ST51SA52SA41ST52SA51 SE101ST49SA52SS41SG117SA41SL512ST52S052ST49SA5 1SL512ST52S052SE53SXS

T123SE60S#S*S&S@SP100SP10SP5SP3SP1SQSPSBSFSISU SZSPSP1SPSPS034S035S036ST49SA50SA41ST50SA50SU5 1SS40SE62SA40SS41SE73ST51S034S045S046S048S048S T49SA50SS39SE75SA39SS41SE86ST51S034S044S047S04 8S048ST49SA51SG53S033ST49SA50SS37SA41SG98SZS04 3S043ST49ST52SA50SS38SG109ST51SA52SA41ST52SA51 SE101ST49SA52SS41SG117SA41SL512ST52S052ST49SA5 1SL512ST52S052SE53SXS

Try this program on NISHIO Hirokazu's EDSAC Simulator http://nhiro.org/learn_language/repos/EDSAC-on-browser/index.html

Enjoy!