Computer Science 3B
Practical Assignment 07
2017-09-28

Deadline - 2017-09-28 17h00 Marks: 40

This practical assignment must be uploaded to eve.uj.ac.za **before** 2017-09-28 17h00. Late or incorrect submissions **will not be accepted**, and will therefore not be marked. You are **not allowed to collaborate** with any other student.

Good coding practices include a proper coding convention and a good use of commenting. Marks will be deducted if these are not present. See the reminder page for more details.

Write an x86 assembly program to determine whether the brackets in a provided equation matches up. Create a ***matching*** function that will do the actual matching. The program should read in a string from the user (reading in strings are explained at the end). The ***matching*** function should then make use of the stack to match the brackets that are present in the string. Round "( )", square "[ ]" and curly "{ }" brackets should be matched. Finally the program should display an appropriate massage to indicate whether the brackets matched up or not.

Example execution:

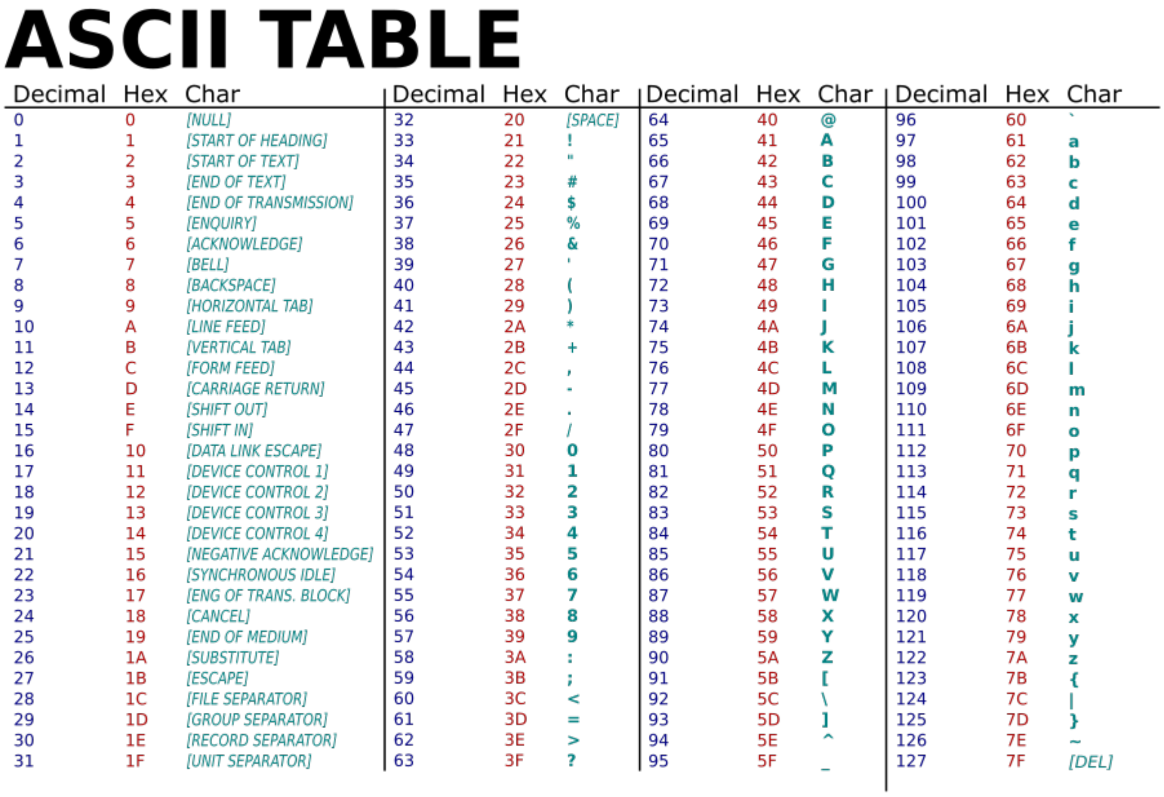
```
Enter string:
x = (23 + z) + [10 - y]
Brackets match

Enter string:
x = (23 + z] + [10 - y]
Brackets do not match

Enter string:
x = {23 + z[ + }10 - y]
Brackets do not match

Enter string:
x = ((23 + z) + [10 - y]
Brackets do not match
```

Reading string values

The IO library contains a function called ***InputStr*** that can be used to read in string values. The ***InputStr*** function takes two parameters namely the *address* where the string should be placed and the *maximum size* of the string. You will thus have to declare an *array* with a maximum size and then pass the address of the array and the size to the ***InputStr*** function. This array can be declared as a global variable.

ASCII table

The ASCII table below will help you to check the brackets correctly. **Remember**: a single character is only one byte! (*HINT*: **EAX** is 32 bits = 4 bytes)



ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Bonus

Make use of a stack allocated array as opposed to a global array.

Mark sheet

1. Design [10]
2. Read equation from user [05]
3. Function ***matching*** [15]
4. Use stack allocated array. [10 (bonus)]
5. Structure and layout (no extra globals, correct data types, no **INVOKE**) [05]
6. Commenting [05]
7. Correct execution. [20]

NB

Submissions which **do not assemble** will be capped at 40%!

Execution marks are awarded for a correctly functioning application and not for having some related code.

Reminder

Your submission must follow the naming convention as set out in the general learning guide.
Example

Surname	Berners-Lee	**Initials**	TJ
Student number	209912345	**Module Code**	CSC3B10
Current Year	2017	**Practical number**	P07

`Berners-Lee_TJ_209912345_ CSC3B10_2017_P07`

Your submission must include the following in a single zip (compressed) file:

- *Source file* (`asm` file) - File containing your solution. Your details must be included at the top of the source code.
- *Program design* (`pdf` file) - File containing your design. Your details must be included at the top of the design.