

O objetivo desta atividade é permitir que o aluno seja capaz de criar um programa em linguagem assembly que manipule caracteres ASCII adequadamente.

### Parte 1 – Tabela ASCII

Observe a tabela ASCII a seguir:

#### Tabela ASCII

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	&#32;	Space	64	40	100	&#64;	@	96	60	140	&#96;	`
1	1	001	SOH (start of heading)	33	21	041	&#33;	!	65	41	101	&#65;	A	97	61	141	&#97;	a
2	2	002	STX (start of text)	34	22	042	&#34;	"	66	42	102	&#66;	B	98	62	142	&#98;	b
3	3	003	ETX (end of text)	35	23	043	&#35;	#	67	43	103	&#67;	C	99	63	143	&#99;	c
4	4	004	EOT (end of transmission)	36	24	044	&#36;	\$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	ENQ (enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	ACK (acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	BEL (bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	BS (backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	TAB (horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	LF (NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	VT (vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	FF (NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	CR (carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m
14	E	016	SO (shift out)	46	2E	056	&#46;	.	78	4E	116	&#78;	N	110	6E	156	&#110;	n
15	F	017	SI (shift in)	47	2F	057	&#47;	/	79	4F	117	&#79;	O	111	6F	157	&#111;	o
16	10	020	DLE (data link escape)	48	30	060	&#48;	0	80	50	120	&#80;	P	112	70	160	&#112;	p
17	11	021	DC1 (device control 1)	49	31	061	&#49;	1	81	51	121	&#81;	Q	113	71	161	&#113;	q
18	12	022	DC2 (device control 2)	50	32	062	&#50;	2	82	52	122	&#82;	R	114	72	162	&#114;	r
19	13	023	DC3 (device control 3)	51	33	063	&#51;	3	83	53	123	&#83;	S	115	73	163	&#115;	s
20	14	024	DC4 (device control 4)	52	34	064	&#52;	4	84	54	124	&#84;	T	116	74	164	&#116;	t
21	15	025	NAK (negative acknowledge)	53	35	065	&#53;	5	85	55	125	&#85;	U	117	75	165	&#117;	u
22	16	026	SYN (synchronous idle)	54	36	066	&#54;	6	86	56	126	&#86;	V	118	76	166	&#118;	v
23	17	027	ETB (end of trans. block)	55	37	067	&#55;	7	87	57	127	&#87;	W	119	77	167	&#119;	w
24	18	030	CAN (cancel)	56	38	070	&#56;	8	88	58	130	&#88;	X	120	78	170	&#120;	x
25	19	031	EM (end of medium)	57	39	071	&#57;	9	89	59	131	&#89;	Y	121	79	171	&#121;	y
26	1A	032	SUB (substitute)	58	3A	072	&#58;	:	90	5A	132	&#90;	Z	122	7A	172	&#122;	z
27	1B	033	ESC (escape)	59	3B	073	&#59;	;	91	5B	133	&#91;	[	123	7B	173	&#123;	{
28	1C	034	FS (file separator)	60	3C	074	&#60;	<	92	5C	134	&#92;	\	124	7C	174	&#124;	
29	1D	035	GS (group separator)	61	3D	075	&#61;	=	93	5D	135	&#93;	]	125	7D	175	&#125;	}
30	1E	036	RS (record separator)	62	3E	076	&#62;	>	94	5E	136	&#94;	^	126	7E	176	&#126;	~
31	1F	037	US (unit separator)	63	3F	077	&#63;	?	95	5F	137	&#95;	_	127	7F	177	&#127;	DEL

Source: [www.LookupTables.com](http://www.LookupTables.com)

A tabela ASCII tem a função de mapear **caracteres** a **códigos de 8 bits** (geralmente representados em seu valor binário, decimal ou hexadecimal).

Na tabela acima podemos ver, por exemplo, que o código ASCII do caractere **"A"** corresponde ao valor **65** (em decimal), ou **41h** (em hexadecimal), que corresponde aos 8 bits **01000001b**.

Da mesma forma, podemos observar que o código ASCII do caractere **"a"** corresponde ao valor **97** (em decimal), ou **61h** (em hexadecimal), que corresponde aos 8 bits **01100001b**.

Já o código ASCII do caractere **"1"** corresponde ao valor **49** (em decimal), ou **31h** (em hexadecimal), que corresponde aos 8 bits **00110001b**.

Observe que nos programas em linguagem assembly:

- **caracteres** são representados entre **aspas simples** ou **aspas duplas**. Ex: 'A' ou "A"
- valores **decimais** são representados por números que podem **opcionalmente** ser terminados com a **letra D** (ou **d**). Ex: 65 ou 65D ou 65d
- valores **hexadecimais** são representados por números **sempre** terminados com a **letra H** (ou **h**). Ex: 41H ou 41h
- valores **binários** são representados por números **sempre** terminados com a **letra B** (ou **b**). Ex: 01000001B ou 01000001b

**Parte 2 - Programa: ATIV02\_1.asm**

- 1) Faça um programa em Assembly x86, que leia uma letra minúscula e a transforme me letra maiúscula.

**O programa deve ter as seguintes mensagens:**

Digite uma letra minúscula: a

A letra maiúscula correspondente eh: A

**Parte 3 - Programa: ATIV2\_2.asm**

Crie um programa em linguagem assembly chamado **ATIV2\_2.asm** que exibe uma mensagem na tela solicitando ao usuário que digite um primeiro número (de 0 a 9), lê o caractere digitado do teclado, exibe uma mensagem na linha seguinte solicitando ao usuário que digite um segundo número (de 0 a 9), lê o caractere digitado do teclado, exibe uma mensagem na linha seguinte informando qual o valor da soma do primeiro com o segundo número e exibe o caractere contendo o resultado da soma.

**OBS:** A soma dos dois números nunca deve ultrapassar o valor 9, ou seja, o usuário sempre deve digitar dois números cuja soma seja menor ou igual a 9.

**Exemplo:**

Digite um primeiro numero: **2**

Digite um segundo numero: **5**

A soma dos dois numeros e: **7**

**ENTREGA**

Cada aluno deve postar os arquivos **ATIV2\_1.asm** e **ATIV2\_2.asm** no CANVAS.