Computer Programming 1

Laboratory 05

(fundamental data types – instructions – part 1)

ASCII encoding

ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(	72	48	110	H	104	68	150	h
9	9	11		41	29	51	)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[	123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	|
29	1D	35		61	3D	75	=	93	5D	135	]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	

1

Exercise 1.1: AND and OR

Write a program that prints the truth table of the AND operator (&&) and the OR operator (||) on the screen.

Steps:

1. Define two boolean variables
2. Initialize the two variables
3. Use the logic operators (&& || !) between the two variables to obtain each row of the tables

Constraint: each row of the table must be the result of a logical expression, i.e., do not use IF-THEN or any other statement to control the flow

AND gate

Input A	Input B	Output
0	0	0
1	0	0
0	1	0
1	1	1

OR gate

Input A	Input B	Output
0	0	0
1	0	1
0	1	1
1	1	1

Exercise 1.2: XOR

Write a program that prints the truth table of the XOR operation on the screen.

Steps:

1. Define two boolean variables
2. Initialize the two variables (the value can be then changed in the program)
3. Use the logic operators (&& || !) between the two variables to obtain each row of the table

Constraint: each row of the table must be the result of a logical expression, i.e., do not use IF-THEN or any other statement to control the flow

Input		Output
A	B	A xor B
0	0	0
0	1	1
1	0	1
1	1	0

2

Example 2.1: VAT

Write a program that defines two real numbers and use them to calculate the customer price of a product by using the following formula.

$$P = P + \frac{P * I}{100}$$

P=price

I=vat

I,P $\in \mathbb{R}$

Constraint: use a precision of 5 digits

Example 2.2: Second-degree equation

Given three real numbers, a, b and c as input, write a program that calculates the solutions of the (complete) second-degree equation (a!=0, $b! = 0, c! = 0$).

$$ax^2 + bx + c = 0$$

$$delta = b^2 - 4ac$$

delta > 0 → $x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

delta < 0 → no solutions

delta =0 → $x = \frac{-b}{2a}$

```
#include <cmath>
 float c = sqrt(5);
```

Reference to the library <cmath>
http://www.cplusplus.com/reference/cmath/

Test: $2x^2 - x - 6 = 0$ → x_1=-1.5 and x_2=2

Test: $x^2 + 3x + 2 = 0$ → x_1=-2 and x_2=-1

Example 2.3(a): Encoding ASCII+10

Given a character as input, write a program that returns the same character in ASCII + 10 encoding

"a" => "k"
"b" => "l"
"c" => "m"
...

Then use -10 from the obtained characters, to turn back to the original character.

Attention to use the casting operator to get the correct value of the variable

Example 2.3(b): Encoding ASCII+10

Given a character as input, write a program that returns the same character in ASCII + 10 encoding

"a" => "k"
"b" => "l"
"c" => "m"

...

Then use -10 from the obtained characters, to turn back to the original character.

Attention to use the casting operator to get the correct value of the variable

Variant: use +/-x, where x is an integer inserted by the user in the standard input, instead of a fixed +/-10 encoding/decoding value

Example 2.4(a): Lower-to-Upper case

Given a lower-case character as input, write a program that stores the character in a variable and that returns the same character in upper-case.

Constraint: do not use the tolower(c) of the cctype library

"a" => "A"

"b" => "B"

"c" => "C"

...

Example 2.4(b): Lower-to-Upper case

Given a lower-case character as input, write a program that stores the character in a variable and that returns the same character in upper-case.

"a" => "A"

"b" => "B"

"c" => "C"

...

Variant: use the function tolower(c) of the cctype library

Example 2.4(c): Lower-to-Upper case

Given a lower-case character as input, write a program that stores the character in a variable and that returns the same character in upper-case.

"a" => "A"
"b" => "B"
"c" => "C"

...

Variant: consider a sequence of X characters, where X is defined by the user and inserted in the standard input

Example 2.4(d): Lower-to-Upper case

Given a lower-case character as input, write a program that stores the character in a variable and that returns the same character in upper-case.

"a" => "A"
"b" => "B"
"c" => "C"

...

Variant: consider a sequence of X characters, where X is defined by the user and inserted in the standard input
Variant: consider as valid only alphabetic lower-case character in input

Example 2.5(a): Variable exchange

Given two variables as input (you can choose their type), a and b, write a program that exchanges the value of a for the value of b and vice versa.

Example 2.5(b): Variable exchange

Given two variables as input (you can choose their type), a and b, write a program that exchanges the value of a for the value of b and vice versa.

Variant: without the use of a temporary variable