Εργαστήριο 4: Είσοδος και Έξοδος Χαρακτήρων

Στο εργαστήριο αυτό, θα ασχοληθούμε με θέματα εισόδου/εξόδου και θα εξοικειωθούμε με τη χρήση συναρτήσεων που μας παρέχει η C για την είσοδο και την έξοδο χαρακτήρων (`getchar` και `putchar`).

Άσκηση 1: Εκτύπωση Χαρακτήρων (printchar.c)

Γράψτε ένα πρόγραμμα `printchar.c` που να εκτυπώνει τους χαρακτήρες με ASCII κωδικό που είναι πολλαπλάσιο του 3, και μεταξύ 33 και 105, μαζί με τους κωδικούς τους σε δεκαδική και δεκαεξαδική μορφή. Το πρόγραμμά σας θέλουμε να τυπώνει σε κάθε γραμμή έναν χαρακτήρα, ακολουθούμενο από τον δεκαδικό και τον δεκαεξαδικό κωδικό του. Παράδειγμα εκτέλεσης ακολουθεί:

```
$ ./printchar
! - (dec: 33, hex: 21)
$ - (dec: 36, hex: 24)
' - (dec: 39, hex: 27)
* - (dec: 42, hex: 2a)
- - (dec: 45, hex: 2d)
0 - (dec: 48, hex: 30)
3 - (dec: 51, hex: 33)
...
```

Παρατηρήστε την έξοδο του προγράμματος σε σχέση με τον παρακάτω πίνακα των ASCII κωδικών:

Char	Dec	Oct	Hex	Char	Dec	Oct	Hex	Char	Dec	Oct	Hex	Char	Dec	Oct	Hex
(nul)	0	000	0x00	(sp)	32	040	0x20	@	64	100	0x40	‘	96	140	0x60
(soh)	1	001	0x01	!	33	041	0x21	A	65	101	0x41	a	97	141	0x61
(stx)	2	002	0x02	”	34	042	0x22	B	66	102	0x42	b	98	142	0x62
(etx)	3	003	0x03	#	35	043	0x23	C	67	103	0x43	c	99	143	0x63
(eot)	4	004	0x04	$	36	044	0x24	D	68	104	0x44	d	100	144	0x64
(enq)	5	005	0x05	%	37	045	0x25	E	69	105	0x45	e	101	145	0x65
(ack)	6	006	0x06	&	38	046	0x26	F	70	106	0x46	f	102	146	0x66
(bel)	7	007	0x07	'	39	047	0x27	G	71	107	0x47	g	103	147	0x67
(bs)	8	010	0x08	(	40	050	0x28	H	72	110	0x48	h	104	150	0x68
(ht)	9	011	0x09	)	41	051	0x29	I	73	111	0x49	i	105	151	0x69
(nl)	10	012	0x0a	*	42	052	0x2a	J	74	112	0x4a	j	106	152	0x6a
(vt)	11	013	0x0b	+	43	053	0x2b	K	75	113	0x4b	k	107	153	0x6b
(np)	12	014	0x0c	,	44	054	0x2c	L	76	114	0x4c	l	108	154	0x6c
(cr)	13	015	0x0d	-	45	055	0x2d	M	77	115	0x4d	m	109	155	0x6d
(so)	14	016	0x0e	.	46	056	0x2e	N	78	116	0x4e	n	110	156	0x6e

Char	Dec	Oct	Hex	Char	Dec	Oct	Hex	Char	Dec	Oct	Hex	Char	Dec	Oct	Hex
(si)	15	017	0x0f	/	47	057	0x2f	O	79	117	0x4f	o	111	157	0x6f
(dle)	16	020	0x10	0	48	060	0x30	P	80	120	0x50	p	112	160	0x70
(dc1)	17	021	0x11	1	49	061	0x31	Q	81	121	0x51	q	113	161	0x71
(dc2)	18	022	0x12	2	50	062	0x32	R	82	122	0x52	r	114	162	0x72
(dc3)	19	023	0x13	3	51	063	0x33	S	83	123	0x53	s	115	163	0x73
(dc4)	20	024	0x14	4	52	064	0x34	T	84	124	0x54	t	116	164	0x74
(nak)	21	025	0x15	5	53	065	0x35	U	85	125	0x55	u	117	165	0x75
(syn)	22	026	0x16	6	54	066	0x36	V	86	126	0x56	v	118	166	0x76
(etb)	23	027	0x17	7	55	067	0x37	W	87	127	0x57	w	119	167	0x77
(can)	24	030	0x18	8	56	070	0x38	X	88	130	0x58	x	120	170	0x78
(em)	25	031	0x19	9	57	071	0x39	Y	89	131	0x59	y	121	171	0x79
(sub)	26	032	0x1a	:	58	072	0x3a	Z	90	132	0x5a	z	122	172	0x7a
(esc)	27	033	0x1b	;	59	073	0x3b	[	91	133	0x5b	{	123	173	0x7b
(fs)	28	034	0x1c	<	60	074	0x3c	\	92	134	0x5c	|	124	174	0x7c
(gs)	29	035	0x1d	=	61	075	0x3d	]	93	135	0x5d	}	125	175	0x7d
(rs)	30	036	0x1e	>	62	076	0x3e	^	94	136	0x5e	~	126	176	0x7e
(us)	31	037	0x1f	?	63	077	0x3f	_	95	137	0x5f	(del)	127	177	0x7f

Άσκηση 2: Κατασκευή Πυραμίδας (pyramid.c)

2.1. Γράψτε ένα πρόγραμμα `pyramid.c` που να εκτυπώνει στην οθόνη σχήμα με την ακόλουθη μορφή χρησιμοποιώντας αποκλειστικά την συνάρτηση `putchar`:

```
*
**
***
****
*****
******
```

Το πλήθος των γραμμών που θα εκτυπωθούν να το διαβάζετε με χρήση `scanf`.

int putchar(int c)
Η συνάρτηση putchar(int c) εκτυπώνει τον χαρακτήρα που αντιστοιχεί στον ASCII κωδικό που δέχεται σαν όρισμα. Ισοδύναμες χρήσεις: putchar('*') ή putchar(42) όπου 42 είναι ο ASCII κωδικός του χαρακτήρα *.

2.2. Τροποποιήστε το `pyramid.c` ώστε να εκτυπώνει στην οθόνη σχήμα με την ακόλουθη μορφή:

```
*
***
*****
*******
*********
***********
```

Extra: τροποποιήστε το `pyramid.c` ώστε ο αριθμός με τον οποίο αυξάνει τον αριθμό των * ανά γραμμή να καθορίζεται επίσης από τον χρήστη με χρήση `scanf`.

Άσκηση 3: Τροποποίηση Κειμένου (lowercase.c)

3.1 Αντιγράψτε το αρχείο `capitalize.c` παρακάτω, μεταγλωττίστε το και εκτελέστε το με είσοδο το ίδιο το πηγαίο αρχείο `capitalize.c`. Παρατηρήστε τη λειτουργία του.

```c
/* File: capitalize.c */

#include <stdio.h>

int main() {
  int ch; /* Be careful! Declare ch as int because of getchar() and EOF */
  ch = getchar(); /* Read first character */
  while (ch != EOF) { /* Go on if we didn\'t reach end of file */
    if (ch >= 'a' && ch <= 'z') { /* If lower case letter */
      ch = ch - ('a'-'A'); /* Move 'a'-'A' positions in the ASCII table */
    }
    putchar(ch); /* Print out character */
    ch = getchar(); /* Read next character */
  }
  return 0;
}
```

Η συνάρτηση `getchar()` διαβάζει έναν χαρακτήρα από την είσοδο και επιστρέφει τον ASCII κωδικό του χαρακτήρα. Αν δεν υπάρχει άλλος χαρακτήρας για διάβασμα στην είσοδο, επιστρέφει την ειδική ακέραια τιμή EOF που είναι ορισμένη στο stdio.h.

3.2 Γράψτε ένα πρόγραμμα `lowercase.c`, ώστε να κάνει το αντίστροφο, δηλαδή να μετατρέπει τους χαρακτήρες που εμφανίζονται με κεφαλαία γράμματα σε χαρακτήρες με μικρά γράμματα.

3.3 Τροποποιήστε το `lowercase.c`, ώστε να μετατρέπονται ταυτόχρονα και οι κεφαλαίοι χαρακτήρες σε μικρούς και οι μικροί σε κεφαλαίους.

Άσκηση 4: Κωδικοποίηση και Αποκωδικοποίηση Κειμένου (encode.c και decode.c)

Σε αυτήν την άσκηση θα γράψουμε προγράμματα για την δεκαεξαδική κωδικοποίηση και αποκωδικοποίηση ενός κειμένου.

4.1 Δημιουργήστε ένα αρχείο κειμένου με όνομα `text` και πληκτρολογήστε σε αυτό ένα τυχαίο κείμενο.

4.2 Γράψτε ένα πρόγραμμα `encode.c` που να διαβάζει από την είσοδο χαρακτήρες και, για καθένα απ' αυτούς, να εκτυπώνει στην έξοδο τον ASCII κωδικό που έχει σε δεκαεξαδική μορφή χρησιμοποιώντας ακριβώς δύο ψηφία. Παράδειγμα εκτέλεσης ακολουθεί:

```
$ /bin/echo hello world | ./encode
68656c6c6f20776f726c640a
```

4.3 Εκτελέστε το πρόγραμμα encode με ανακατεύθυνση εισόδου από το αρχείο `text`.

4.4 Εκτελέστε το πρόγραμμα encode με ανακατεύθυνση εισόδου από το αρχείο `text` και ανακατεύθυνση εξόδου στο αρχείο `encoded_text`.

4.5 Γράψτε ένα πρόγραμμα `decode.c` που να διαβάζει από την είσοδο τα δεκαεξαδικά ψηφία για τον κάθε χαρακτήρα και για κάθε δύο από αυτά τα ψηφία να εκτυπώνει τον χαρακτήρα που αντιστοιχεί σε αυτόν τον δεκαεξαδικό αριθμό. Παράδειγμα εκτέλεσης ακολουθεί:

```
$ /bin/echo -ne 68656c6c6f20776f726c640a | ./decode
hello world
```

Ο αριθμός `68` αντιστοιχεί στον χαρακτήρα `'h'` κοκ. Για την αποκωδικοποίηση συνιστούμε να χρησιμοποιήσετε την συνάρτηση `getchar()`. Αν το επιθυμείτε μπορείτε να δοκιμάσετε και άλλες λύσεις όπως π.χ., την `scanf("%02x", ...)` αλλά μην ξεχάσετε να αποθηκεύσετε το αποτέλεσμα σε μεταβλητή τύπου `int` (και να μεταγλωττίσετε με το flag `-Wall` αλλιώς είναι πιθανό να σας ξεφύγουν διάφορα λαθάκια).

4.6 Μεταγλωττίστε το και εκτελέστε το με ανακατεύθυνση της εισόδου από το αρχείο `encoded_text`.

4.7 Εκτελέστε το πρόγραμμα `decode` για την αποκωδικοποίηση ενός κωδικοποιημένου κειμένου, έτσι ώστε να παίρνει την είσοδό του απ' ευθείας από το πρόγραμμα `encode`, το οποίο να καλείται έτσι ώστε να κωδικοποιεί το αρχείο `text`, χωρίς να χρησιμοποιήσετε το ενδιάμεσο αρχείο `encoded_text`. Παράδειγμα εκτέλεσης ακολουθεί:

```
$ /bin/echo hello world | ./encode | ./decode
hello world
```