Α' Εξάμηνο
ΕΙΣΑΓΩΓΗ ΣΤΟ ΔΟΜΗΜΕΝΟ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ

Εργαστήριο – 6η εβδομάδα.
Κοζάνη, 19 Νοεμβρίου 2008.

Σε αυτή την ομάδα ασκήσεων συνδυάζουμε τις μέχρι τώρα γνώσεις μας από τις δομές επανάληψης και επιλογής, τις συναρτήσεις, τους πίνακες και τα αλφαριθμητικά για να κάνουμε εφαρμογές με ελεγχόμενη έξοδο στην οθόνη.

Άσκηση p6-1
Κατ' αρχήν θέλουμε να δούμε αν μπορούμε να ανανεώνουμε την οθόνη σβήνοντας τα παλιά δεδομένα και δείχνοντας σε κάθε επανάληψη μόνο τα νέα. Δοκιμάζουμε να γράψουμε ένα πρόγραμμα που αυξάνει σταδιακά έναν αριθμό και δείχνει κάθε φορά μόνο την καινούρια τιμή αφού "καθαρίσει" προηγουμένως την οθόνη.

Γράψτε το ακόλουθο πρόγραμμα σε ένα αρχείο με το όνομα π.χ. p6_1.c, μεταγλωττίστε το και εκτελέστε το:

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char **argv)
{
    int s, t, kount=0;

    while(1) {
        system("clear");
        printf("\n\n\n\n\n\t\t\ttime=%d\n", kount);
        for(t=0; t < 1000; t++)
            for(s=0; s<10000; s++);
        kount++;
        if(kount > 999)
            break;
    }
    return 0;
}
```

Ερωτήσεις:
Τι έξοδο παίρνετε;
Τι κάνει η εντολή while(1) { ... } και πώς φεύγουμε από αυτή τη δομή επανάληψης;
Τι ρόλο παίζουν τα δύο for μέσα στη while;
Τι κάνει η εντολή system;

Άσκηση p6-2
Εδώ βελτιώνουμε λίγο το προηγούμενο πρόγραμμα για να του υπαγορεύουμε το ρυθμό ανανέωσης της οθόνης με έναν αριθμό από τη γραμμή εντολών.

Γράψτε το ακόλουθο πρόγραμμα σε ένα αρχείο με το όνομα π.χ. p6_2.c, και μεταγλωττίστε το. Με γαλάζιο είναι ο κώδικας που παραμένει ίδιος με πριν και με κόκκινο τα νέα στοιχεία ή οι αλλαγές.

```
#include <stdio.h>
#include <stdlib.h>

#define DELAY 25

int main(int argc, char **argv)
{
    int delay, s, t, kount=0;

    delay = (argv[1]?atoi(argv[1]):DELAY);
    while(1) {
        system("clear");
        printf("\n\n\n\n\n\t\t\ttime=%d\n", kount);
        for(t=0; t<1000; t++)
            for(s=0; s<1000*delay; s++);
        if(kount++ > 999)
            break;
    }

    return 0;
}
```
Μετά εκτελέστε το δίνοντας του διαφορετικά αριθμητικά ορίσματα από τη γραμμή εντολών: 5, 10, 20, 50, 100. Δοκιμάστε και χωρίς ορίσματα.

Ερωτήσεις:
Τι έξοδο παίρνετε;
Τι σημαίνει η οδηγία #define;
Τι σημαίνουν τα ορίσματα της main στη γραμμή int main(int argc, char **argv) ;
Τι κάνει η συνάρτηση atoi ;
Τι κάνει ο τριαδικός τελεστής (? :) ;
Τι ρόλο φαίνεται να παίζει η μεταβλητή delay στο δεύτερο for ;

Άσκηση p6-3 (μεταγλώττιση και εκτέλεση)
Τροποποιούμε λίγο την εικόνα που εμφανίζεται στην οθόνη. Εκτός από το μετρητή, θα εμφανίζεται ένα σημείο (σύμβολο '*') που θα μετακινείται γραμμικά πάνω στην οθόνη.

Γράψτε το επόμενο πρόγραμμα σε ένα αρχείο με το όνομα π.χ. p6_3.c, και μεταγλωττίστε το. Και πάλι, με γαλάζιο είναι ο κώδικας που παραμένει ίδιος με πριν και με κόκκινο τα νέα στοιχεία ή οι αλλαγές.

```c
#include <stdio.h>
#include <stdlib.h>

#define DELAY 25
#define OFFSET 10

int main(int argc, char **argv)
{
    int i, delay, s, t, kount=0;

    delay = (argv[1]?atoi(argv[1]):DELAY);
    while(1) {
        system("clear");
        printf("\ntime=%d\n\n\n", kount);
        for(i=0; i<kount+OFFSET; i++)
            putchar(' ');
        putchar('*');
        fflush(stdout);

        for(t=0; t<1000; t++)
            for(s=0; s<1000*delay; s++);
        if(kount++ > 999)
            break;
    }

    return 0;
}
```

Μετά εκτελέστε το δίνοντας του διαφορετικά αριθμητικά ορίσματα από τη γραμμή εντολών, όπως και προηγουμένως.

Ερωτήσεις:
Τι έξοδο παίρνετε τώρα;
Τι κάνει η συνάρτηση puchar;
Τι κάνει η συνάρτηση fflush;

Άσκηση p6-4 (μεταγλώττιση και εκτέλεση)
Τροποποιούμε και άλλο την εικόνα που εμφανίζεται στην οθόνη. “Διώχνουμε” τον κέρσορα στην άκρη και κάνουμε το σύμβολο * να επανέρχεται περιοδικά.

Γράψτε το επόμενο πρόγραμμα σε ένα αρχείο με το όνομα π.χ. p6_4.c, και μεταγλωττίστε το. Και πάλι, με γαλάζιο είναι ο κώδικας που παραμένει ίδιος με πριν και με κόκκινο τα νέα στοιχεία ή οι αλλαγές.

```c
#include <stdio.h>
#include <stdlib.h>

#define DELAY  25
#define OFFSET 10
#define WIDTH  80

int main(int argc, char **argv)
{
    int i, delay, s, t, kount=0;

    delay = (argv[1]?atoi(argv[1]):DELAY);
    while(1) {
        system("clear");
        printf("\ntime=%d\n\n\n", kount);
        for(i=0; i<OFFSET; i++)
            putchar(' ');
        for(i=0; i<WIDTH; i++)
            putchar(i==kount%WIDTH?'*':' ');
        putchar(' ');
        fflush(stdout);

        for(t=0; t<1000; t++)
            for(s=0; s<1000*delay; s++);
        if(kount++ > 999)
            break;
    }

    return 0;
}
```

Μετά εκτελέστε το δίνοντας του διαφορετικά αριθμητικά ορίσματα από τη γραμμή εντολών, όπως και προηγουμένως.

Ερωτήσεις:
Τι έξοδο παίρνετε τώρα;
Τι κάνει ο τελεστής %;

Άσκηση p6-5 (μεταγλώττιση και εκτέλεση)
Τώρα υλοποιούμε μια άλλη περιοδική κίνηση: το σημείο που παριστάνουμε * θα “συγκρούεται” σε κάποια “τοιχώματα” και θα πηγαίνει μπρος-πίσω, σα μπαλάκι του τέννις.

Γράψτε το επόμενο πρόγραμμα σε ένα αρχείο με το όνομα π.χ. p6_5.c, και μεταγλωττίστε το.

```
#include <stdio.h>
#include <stdlib.h>

#define DELAY  25
#define OFFSET 10
#define WIDTH  80

int main(int argc, char **argv)
{
    int i, delay, s, t, kount=0;

    delay = (argv[1]?atoi(argv[1]):DELAY);
    while(1) {
        system("clear");
        printf("\ntime=%d\n\n\n", kount);
        for(i=0; i<OFFSET-1; i++)
            putchar(' ');
        putchar('|');
        for(i=0; i<WIDTH; i++)
            if(kount/WIDTH%2)
                putchar(i==WIDTH-kount%WIDTH?'*':' ');
            else
                putchar(i==kount%WIDTH?'*':' ');
        putchar('|');
        fflush(stdout);

        for(t=0; t<1000; t++)
            for(s=0; s<1000*delay; s++);
        if(kount++ > 999)
            break;
    }

    return 0;
}
```

Μετά εκτελέστε το δίνοντας του διαφορετικά αριθμητικά ορίσματα από τη γραμμή εντολών, όπως και προηγουμένως.

Ερωτήσεις:
Τι έξοδο παίρνετε τώρα;
Παρατηρείστε και εξηγείστε πώς ο τριαδικός τελεστής υπεισέρχεται στο όρισμα της putchar για να δώσει το επιθυμητό αποτέλεσμα.

Άσκηση p6-6 (μεταγλώττιση και εκτέλεση)

Κάνουμε μια μικρή τροποποίηση στην προηγούμενη εικόνα: το μπαλάκι κινείται τώρα μέσα σε ένα "κουτί" που φτιάχνουμε σχεδιάζοντας δύο οριζόντιες γραμμές, πάνω και κάτω από την ευθεία στην οποία κινείται.
Το πρόγραμμα όμως υφίσταται και ορισμένες σημαντικές ανακατατάξεις αφού εισάγουμε μια σειρά από συναρτήσεις για να ελέγχουμε καλύτερα την ανάπτυξη της κάθε λειτουργίας, καθώς και για να έχουμε την εποπτεία της δομής του βλέποντας τις κλήσεις των συναρτήσεων που υπάρχουν στη main.

Γράψτε το επόμενο πρόγραμμα σε ένα αρχείο με το όνομα π.χ. p6_6.c, και μεταγλωττίστε το. Εδώ υπάρχουν μεγάλες αλλαγές σε σχέση με το προηγούμενο (όπως πάντα, με κόκκινο χρώμα). Εκτελέστε όπως πριν.

```
#include <stdio.h>
#include <stdlib.h>

#define DELAY  25
#define OFFSET 10
#define WIDTH  100

void draw_line(void);
void draw_point(int);
void time_delay(int, int*);

int main(int argc, char **argv)
{
    int i, delay, kount=0;

    delay = (argv[1]?atoi(argv[1]):DELAY);
    while(1) {
        system("clear");
        printf("\ntime=%d\n\n\n", kount);

        draw_line();
        draw_point(kount);
        draw_line();
        fflush(stdout);

        time_delay(delay, &kount);
    }
    return 0;
}

void draw_line( void )
{
```

```
    int i;
    putchar('\n');
    for(i=0; i<OFFSET-1; i++)
        putchar(' ');
    putchar('|');
    for(i=0; i<WIDTH; i++)
        putchar('-');
    putchar('|');
    putchar('\n');
    return;
}


void draw_point( int k )
{
    int i;
    for(i=0; i<OFFSET-1; i++)
        putchar(' ');
    putchar('|');
    for(i=0; i<WIDTH; i++)
        if(k/WIDTH%2)
            putchar(i==WIDTH-k%WIDTH?'*':' ');
        else
            putchar(i==k%WIDTH?'*':' ');
    putchar('|');
    return;
}

void time_delay(int delay, int *k)
{
    int s, t, stat;
    for(t=0; t<1000; t++)
        for(s=0; s<1000*delay; s++);
    if((*k)++ > 1000)
       exit(stat);
    return;
}
```

Ερωτήσεις:
Τι έξοδο παίρνετε τώρα;
Τι είναι πρωτότυπο μιας συνάρτησης και πού χρειάζεται;
Τι είναι πέρασμα παραμέτρου (συνάρτησης) κατά τιμή και τι είναι πέρασμα κατά διεύθυνση;
Εντοπίστε τη μεταβλητή ή τις μεταβλητές που περνάνε κατά διεύθυνση ως ορίσματα συναρτήσεων.
Και μόνο από τα ονόματα των συναρτήσεων καταλαβαίνουμε αρκετά για τη λειτουργία τους. Σχολιάστε τη

σημασία του ευανάγνωστου και κατανοητού κώδικα.

Άσκηση p6-7
Κάνουμε ένα τελευταίο "κόλπο" μετατρέποντας την κίνηση από ευθύγραμμη σε τεθλασμένη (ζικ-ζακ). Εδώ φαίνεται το κέρδος από την ανακατάταξη που κάναμε προηγουμένως αφού τροποποιούμε μόνο μία συνάρτηση που αφήνει ανέπαφες τις υπόλοιπες. Η δε αλλαγή που κάνουμε, συνίσταται στην εισαγωγή ενός πίνακα χαρακτήρων ο οποίος παριστάνει το τμήμα της οθόνης που θέλουμε να σχεδιάσουμε. Κάθε φορά τοποθετούμε τον αστερίσκο στο κατάλληλο στοιχείο του πίνακα και έτσι παίρνουμε ένα στιγμιότυπο το οποίο μετά δεν έχουμε παρά να εμφανίσουμε στην οθόνη τυπώνοντας τις γραμμές του πίνακα ως αλφαριθμητικά με τη βοήθεια της printf.

Γράψτε το επόμενο πρόγραμμα σε ένα αρχείο με το όνομα π.χ. p6_7.c, και μεταγλωττίστε το. Εκτελέστε όπως πριν.

```
#include <stdio.h>
#include <stdlib.h>

#define DELAY  25
#define OFFSET 10
#define WIDTH  100
#define DEPTH  10

void draw_line(void);
void draw_point(int);
void time_delay(int, int*);

int main(int argc, char **argv)
{
    int i, delay, kount=0;

    delay = (argv[1]?atoi(argv[1]):DELAY);
    while(1) {
        system("clear");
        printf("\ntime=%d\n\n\n", kount);

        draw_line();
        draw_point(kount);
        draw_line();
        fflush(stdout);

        time_delay(delay, &kount);
    }
    return 0;
}
```

```
void draw_line( void )
{
    int i;
    for(i=0; i<OFFSET-1; i++)
        putchar(' ');
    putchar('|');
    for(i=0; i<WIDTH; i++)
        putchar('-');
    putchar('|');
    putchar('\n');
    return;
}


void draw_point( int k )
{
    int i, j;
    char places[DEPTH][WIDTH+OFFSET+2];

    for(i=0; i<DEPTH; i++)
        for(j=0; j<WIDTH+OFFSET+2; j++)
            places[i][j] = ' ';
    for(i=0; i<DEPTH; i++) {
        places[i][OFFSET-1] = '|';
        places[i][WIDTH+OFFSET] = '|';
        places[i][WIDTH+OFFSET+1] = '\0';
    }
    if(k/WIDTH%2)
        j = OFFSET+WIDTH-k%WIDTH;
    else
        j = OFFSET+k%WIDTH;
    if(k/DEPTH%2)
        i = DEPTH-k%DEPTH;
    else
        i = k%DEPTH;
    places[i][j] = '*';
    for(i=0; i<DEPTH; i++)
        printf("%s\n", places[i]);
    return;
}

void time_delay(int delay, int *k)
```

```
{
    int s, t, stat;
    for(t=0; t<1000; t++)
        for(s=0; s<1000*delay; s++);
    if((*k)++ > 1000)
       exit(stat);
    return;
}
```

Ερωτήσεις:
Τι έξοδο παίρνετε τώρα;
Τι ρόλο παίζει ο μηδενικός χαρακτήρας, '\0', σε ένα πίνακα χαρακτήρων;