

1 Ανάγνωση (20%)

Τι θα τυπώσει το παρακάτω πρόγραμμα:

```
1 #include<stdio.h>
2 void f() {
3     static int i = 0;
4     printf("%d ", i++);
5 }
6 void g() {
7     int j = 42;
8     f();
9     printf("%d ", j++);
10    f();
11 }
12 int main() {
13     g();
14     g();
15 }
```

2 Πίνακες (20%)

Υποθέστε την δήλωση του (global) πίνακα: int Array[100];
Γράψτε μια συνάρτηση σε C, η οποία να τυπώνει όλα τα στοιχεία του πίνακα με printf, με αριθμητική δεικτών, δηλαδή χωρίς να χρησιμοποιήσετε τα σύμβολα '[' και ']'.

3 Λίστες (20%)

Υποθέστε μια απλά συνδεδεμένη λίστα που ξεκινά από το δείκτη head:

```
1 struct list {  
2     int data;  
3     struct list *next;  
4 };  
5 struct list *head;
```

Γράψτε ένα for loop το οποίο να τυπώνει το πεδίο data από όλα τα στοιχεία της απλά συνδεδεμένης λίστας, ξεκινώντας από το δείκτη head. Μπορείτε να χρησιμοποιήσετε την printf.

4 Debug (20%)

Ο παρακάτω κώδικας περιέχει ένα λάθος χρόνου εκτέλεσης. Εξηγήστε ποιό είναι, και πώς διορθώνεται.

```
1 #include<stdio.h>  
2 int main() {  
3     char *msg = "Snaggletooth";  
4     int len = strlen(msg);  
5     for(int i = 0; i <= len; i++) {  
6         msg[i] += 1;  
7     }  
8     printf("%s has length %d\n", msg, strlen(msg));  
9 }
```

5 Rotation (20%)

Γράψτε μια συνάρτηση rot1, η οποία να δέχεται έναν πίνακα ακεραίων και το μήκος του πίνακα ως ορίσματα, και μετακινεί όλα τα στοιχεία του πίνακα στην επόμενη θέση, και το τελευταίο στοιχείο στην πρώτη θέση του πίνακα. Για παράδειγμα, για τον πίνακα:

```
1 int a[5] = {1, 2, 3, 4, 5};  
2 rot1(a, 5);
```

Μετά την κλήση της συνάρτησης, ο πίνακας a θα περιέχει {5, 1, 2, 3, 4}.

6 Files (20%)

Γράψτε ένα πρόγραμμα C το οποίο ανοίγει ένα αρχείο κειμένου και τυπώνει τα περιεχόμενα του αρχείου στο stdout. Μπορείτε να χρησιμοποιήσετε τις συναρτήσεις fopen, fclose, fgetc, putchar, και τη σταθερά EOF από τη βιβλιοθήκη stdio.h. Το όνομα του αρχείου κειμένου θα δίνεται από τη γραμμή εντολών.

5 Μεταβλητές (20%)

Τι θα τυπώσει το παρακάτω πρόγραμμα;

```
1 #include<stdio.h>
2 int f(int);
3 int g(int);
4
5 int main() {
6     for(int i = 0; i < 4; i++) {
7         f(g(10-i));
8     }
9 }
10
11 int g(int i) {
12     static int x = 0;
13     printf("x = %d, i = %d\n", x, i);
14     return x++;
15 }
16
17 int f(int y) {
18     printf("y = %d\n", y);
19 }
```

6 (30%)

Γράψτε σε C μια συνάρτηση `find_next_max` η οποία δέχεται ένα πίνακα ακεραίων και μέγεθος του πίνακα, και επιστρέφει τη θέση στην οποία βρίσκεται ο δεύτερος μεγαλύτερος ακέραιος.

2 Strings (15%)

Να γράψετε μια συνάρτηση `read_words` σε γλώσσα C, η οποία θα δέχεται τρία ορίσματα: έναν ακέραιο αριθμό N , έναν πίνακα χαρακτήρων A , μεγέθους τουλάχιστον N , και έναν ακέραιο αριθμό W . Η συνάρτησή σας θα πρέπει χρησιμοποιώντας τη συνάρτηση `scanf` να διαβάζει το πολύ W λέξεις από το πληκτρολόγιο και να τις αποθηκεύει στον πίνακα A , χωρίς κενά μεταξύ τους. Αν διαβάσει W λέξεις, ή αν διαβάζοντας τις λέξεις φτάσει στους $N-1$ συνολικά χαρακτήρες, θα σταματά, θα τερματίζει σωστά τον πίνακα ώστε να περιέχει αλφαριθμητικό, και θα επιστρέψει. Η συνάρτησή σας δεν θα έχει τιμή επιστροφής, αλλά θα κάνει όλες τις αλλαγές που χρειάζονται στον πίνακα A .

3 Πίνακες (20%)

Τι θα τυπώσει το παρακάτω πρόγραμμα;

```
1 #include<stdio.h>
2 int main() {
3     char *s[5] = { "For", "whom", "the", "bell", "tolls" };
4     int i;
5     char **p = s;
6
7     for(i = 4; i >= 0; i--) {
8         printf("%c%c", (*p)[4-i], (i != 0 ? '-' : '\n'));
9         p++;
10    }
11 }
```

4 Debugging (20%)

Βρείτε, αιτιολογήστε και διορθώστε τα συντακτικά λάθη και τα λάθη χρόνου εκτέλεσης στο παρακάτω πρόγραμμα, κάνοντας τις ελάχιστες δυνατές αλλαγές:

```
1 #include<stdio.h>
2 int main() {
3     char arr[] = "Eleanor Rigbe";
4     char **p;
5
6     arr[14] = 'y';
7
8     for (p = arr; *p != '\0'; p++) {
9         printf("%s\n", p);
10    }
11 }
```

ΗΥ-100: Εισαγωγή στην Επιστήμη Υπολογιστών
Διαγώνισμα Προόδου

Οδηγίες

Διαβάστε προσεκτικά ολόκληρο το διαγώνισμα και σκεφτείτε το χρόνο που θα αφιερώσετε στο κάθε ερώτημα. Λάβετε υπόψη το σχετικό βάρος του κάθε ερωτήματος στο συνολικό βαθμό.

Το διαγώνισμα γίνεται με **κλειστό βιβλίο και κλειστές σημειώσεις**. Δεν επιτρέπεται να ανατρέξετε σε βιβλίο, σημειώσεις ή άλλο συλικό κατά τη διάρκεια της εξέτασης. Δεν επιτρέπεται επίσης η χρήση αριθμομηχανών, τηλεφώνων και άλλων ηλεκτρονικών συσκευών.

Στις ερωτήσεις όπου χρειάζεται να γράψετε κώδικα, δεν πρέπει να χρησιμοποιήσετε οποιαδήποτε συνάρτηση βιβλιοθήκης, εκτός αν αναφέρεται ρητά στο ερώτημα.

Μόλις τελειώσετε, παραδώστε το συμπληρωμένο διαγώνισμα και το αποδεικτικό ταυτότητας στους διδάσκοντες ή στους επιτηρητές στην έδρα. Αν φεύγετε πριν το τέλος της εξέτασης, παρακαλείστε να μην κάνετε θόρυβο για να μην ενοχλείτε τους συμφοιτητές σας που γράφουν.

Ονοματεπώνυμο:

Αριθμός μητρώου:

1 Ανάγνωση κώδικα (15%)

Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
1 #include<stdio.h>
2 void mix(int *pt1, int *ptr2, int x);
3
4 int main() {
5     int i = 1, j = 2, k = 3;
6
7     mix(&i, &j, k);
8     printf("%d %d %d\n", i, j, k);
9 }
10
11 void mix(int 100 200 300 *pt1, int *ptr2, int x) {
12     ptr1 = ptr2;
13     *ptr1 = 100;
14     *ptr2 = 200;
15     x = (*ptr1) + (*ptr2);
16     printf("%d\n", x);
17 }
```

Ερώτηση	Ισχυρότητα
1	
2	
3	
4	
5	
6	
Σύνολο	

Μη γράψετε στον παραπάνω πίνακα.

```

Int main() {
    char str1[] = "bar";
    char str2[] = "foo";
    myswap(str1, str2);
    printf("%s %s", str1, str2);
    return 0;
}

```

4 Αναδρομή (40%)

Να γράψετε μια αναδρομική συνάρτηση `print_reverse(int N)` η οποία θα διαβάζει Ν λέξεις μήκους το πολύ 100 χαρακτήρων, με τη συνάρτηση `scanf ("%s", ...)` και θα τις τυπώνει αντίστροφη σειρά από αυτή που διαβάστηκαν. Χρησιμοποιήστε μνήμη εντός του stack frame της συνάρτησης και αναδρομή για να αποθηκεύσετε τις λέξεις, και όχι δυναμική δέσμη μνήμης.

5 Ανάγνωση κώδικα (20%)

Τι θα τυπώσει το παρακάτω πρόγραμμα:

```

#include<stdio.h>
void main() {
    int s[6] = { 2, 4, 5, 0, 3, 1 };
    int i, j;

    j = 0;
    for(i = 0; i <= 10; i++) {
        printf("%d ", s[j]);
        j = s[i];
    }
}

```

1 Πίνακες (20%)

Τι περιέχει ο πίνακας αριστερά όταν τελειώσει το πρόγραμμα;

```
int main() {
    int i, a[10] = {0};
    for(i = 0; i < 10; i++) {
        a[i++ ] = 20;
    }
    return 0;
}
```

2 Αλφαριθμητικά (20%)

Τι θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>
int main() {
    char str[] = "question";
    str[5] = '0';
    printf("%s\n", str);
}
```

3 Κλήση συνάρτησης (20%)

Τι θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>

void myswap(char *str1, char *str2) {
    char temp = *str1;
    *str1 = *str2;
    *str2 = temp;
}
```

HY-100: Εισαγωγή στην Επιστήμη Υπολογιστών

Επαναληπτικό Διαγώνισμα

Όνοματεπώνυμο: _____ Α.Μ.: _____

1 Ανάγνωση Κώδικα (20%)

Τί κάνει η παρακάτω συνάρτηση;

```
void fun(int num) {
    if (num / 8) fun(num / 8);
    putchar(num % 8 + '0');
}
```

2 Debugging (40%)

Δίνεται το παρακάτω πρόγραμμα. Η συνάρτηση concat ενώνει δύο αλφαριθμητικά, αντιγράφοντας έναν έναν τους χαρακτήρες του δεύτερου στο τέλος του πρώτου. Επομένως, η αναμενόμενη έξοδος του προγράμματος είναι "Hello world!". Το πρόγραμμα μεταγλωττίζεται χωρίς λάθη μεταγλωττισης, αλλά τερματίζεται με segmentation fault όταν εκτελείται. Εξηγήστε γιατί.

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

void concat(char *a, char *b) {
    int i, m, n;
    m = strlen(a);
    n = strlen(b);
    for (i = 0; i <= n; i++) {
        a[m+i] = b[i];
    }
}

int main() {
    char *str1 = "Hello ";
    char *str2 = "world!";
    concat(str1, str2);
    printf("%s", str1);
    return 0;
}
```

3 Αναδρομή (40%)

Να γράψετε μια αναδρομική συνάρτηση η οποία θα υπολογίζει το παραγοντικό του x (γινόμενο από 1 μέχρι x), η οποία να έχει τον τύπο:

unsigned int factorial(unsigned int x);

Μη χρησιμοποιήστε επανάληψη, στατικές ή καθολικές μεταβλητές, αλλά μόνο αναδρομή.

ΥΥ-100: Εισαγωγή στην Επιστήμη Υπολογιστών

Επαναληπτικό Διαγώνισμα

Όνοματεπώνυμο: _____ Α.Μ.: _____

1 Ανάγνωση Κώδικα (20%)

Τί κάνει η παρακάτω συνάρτηση:

```
void fun(int num) {
    if (num / 8) fun(num / 8);
    putchar(num % 8 + '0');
}
```

2 Debugging (40%)

Δίνεται το παρακάτω πρόγραμμα. Η συνάρτηση concat ενώνει δύο αλφαριθμητικά, αντιγράφοντας έναν έναν τους χαρακτήρες του δεύτερου στο τέλος του πρώτου. Επομένως, η αναμένόμενη έξοδος του προγράμματος είναι "Hello world!". Το πρόγραμμα μεταγλωττίζεται χωρίς λάθη μεταγλώττισης, αλλά τερματίζει με segmentation fault όταν εκτελείται. Εξηγήστε γιατί.

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

void concat(char *a, char *b) {
    int i, m, n;
    m = strlen(a);
    n = strlen(b);
    for (i = 0; i <= n; i++) {
        a[m+i] = b[i];
    }
}

int main() {
    char *str1 = "Hello ";
    char *str2 = "world!";
    concat(str1, str2);
    printf("%s", str1);
    return 0;
}
```

3 Αναδρομή (40%)

Να γράψετε μια αναδρομική συνάρτηση η οποία θα υπολογίζει το παραγοντικό του x (γινόμενο από 1 μέχρι x), η οποία να έχει τον τύπο:

unsigned int factorial(unsigned int x);

Μη χρησιμοποιήστε επανάληψη, στατικές ή καθολικές μεταβλητές, αλλά μόνο αναδρομή.

3 Out of the box (10%, bonus)

Σε ποιά γραμμή του παρακάτω προγράμματος θα εμφανίσει λάθος μεταγλώττισης ο gcc και γιατί;

```
1 #include<stdio.h>
2 #define CIRCUM(R) (3.14*R*R);
3 int main()
4 {
5     float r=1.0,c;
6     c= CIRCUM(r);
7     printf("\n%.f",c);
8     If(CIRCUM(r)==6.28)
9         printf("\nGobbledygook");
10    return 0;
11 }
```

3

4. 4

5. 0

1.7 Τι θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>

int main() {
    int i;
    for(i = 0; i < 5; i++) {
        int i = 10;
        printf("%d", i);
        i++;
    }
    return 0;
}
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. 10 11 12 13 14
4. 0 1 2 3 4
5. 1 2 3 4 5
6. 10 10 10 10 10

10

1.8 Array Placement

Υποθέστε τον ακόλουθο πίνακα 2 διαστάσεων:

```
char A[100][100];
```

Υποθέστε ότι η κύρια μνήμη μπορεί να προσπελάσει κάθε 1 byte που χωρά ακριβώς 1 char, και ότι ο πίνακας ξεκινά στη διεύθυνση 0. Σε ποιά διεύθυνση είναι αποθηκευμένο το στοιχείο A[40][50] του πίνακα;

1. 4040
2. 5040
3. 4050
4. 5050

1.3 Οι παρακάτω δύο εντολές είναι ισοδύναμες;

```
p = ptr;  
p = *(&ptr);
```

1. Ναι

2. Όχι

1.4 Ποιός από τους παρακάτω χαρακτήρες βρίσκεται στο τέλος κάθε αλφαριθμητικού στην C;

1. '\0'

2. "\0"

3. '0'

4. "\n"

1.5 Ποιά θα είναι η τιμή της μεταβλητής x στο τέλος του παρακάτω προγράμματος;

```
int main() {  
    int x = 0;  
    for(x = 0; x < 20; x++) { }  
}
```

1. 0

2. 1

3. 19

4. 20

5. 21

1.6 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>  
  
int main() {  
    const int i = 3;  
    printf("%d", ++i);  
    return 0;  
}
```



Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση

2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)

1.9 Τί θα τυπώσει το παρακάτω πρόγραμμα;

Υποθέστε ότι ο πίνακας arr ξεκινά στη διεύθυνση 2000 και ότι το μέγεθος ενός int είναι 32 bit.
Η συνάρτηση printf με το αλφαριθμητικό μορφοποίησης "%u" τυπώνει unsigned int αριθμούς
στο δεκαδικό σύστημα αριθμησης.

```
#include <stdio.h>
int main() {
    int arr[5];
    /* Assume that base address of arr is 2000 and size of integer
       is 32 bit */
    arr++;
    printf("%u", arr);
}
return 0;
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. 2000
4. 2002
5. 2004
6. 2016

1.10 Τί κάνει η παρακάτω συνάρτηση;

```
void fun(int num) {
    if (num / 8) fun(num / 8);
    putchar(num % 8 + '0');
}
```

1. Μετρά τον αριθμό bits που είναι 1 στην παράμετρο num
2. Τυπώνει την παράμετρο num στο δυαδικό σύστημα
3. Τυπώνει την παράμετρο num στο οκταδικό σύστημα
4. Αθροίζει όλα τα bits στην παράμετρο num
5. Τυπώνει το υπόλοιπο της διαίρεσης της παραμέτρου num με το 8

1.15 Τι θα τυπώσει το παρακάτω πρόγραμμα;

```
#include <stdio.h>
int main()
{
    int (*ptr)(int) = fun;
    (*ptr)(3);
    return 0;
}
int fun(int n)
{
    for(;n > 0; n--)
        printf("12345");
    return 0;
}
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. 12345 12345 12345 12345
4. 12345 12345 12345¹
5. 12345 12345

4. Ace

5. eca

6. sedapS fO ecA

1.14 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>
struct foo {
    char a;
    float b;
};

struct bar {
    int a[3];
    char b;
    struct foo f;
};

int main() {
    struct bar v = {{1,2,3}, 'C', {'a', 3.1}};
    printf("%d, %2f", v.a[2], v.f.b);
    return 0;
}
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. 2, 3.100000
4. 3, 3.10
5. 3 4
6. 2, 3.1
7. 3, 'a'

1.12 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>

void swap(char *str1, char *str2) {
    char *temp = str1;
    str1 = str2;
    str2 = temp;
}

int main() {
    char *str1 = "bar";
    char *str2 = "foo";
    swap(str1, str2);
    printf("%s %s", str1, str2);
    return 0;
}
```

- ① Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. bar foo
4. foo bar
5. foo foo
6. bar bar

1.13 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>

void f(char *s) {
    if (*s != '\0' && *s != ' ') {
        f(s + 1);
        putchar(*s);
    }
}

int main() {
    f("Ace Of Spades");
    return 0;
}
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. Ace Of Spades

HY-100: Εισαγωγή στην Επιστήμη Υπολογιστών

Τελικό Διαγώνισμα

Οδηγίες

Διαβάστε προσεκτικά ολόκληρο το διαγώνισμα και σκεφτείτε το χρόνο που θα αφιερώσετε στο κάθε ερώτημα. Λάβετε υπόψη το σχετικό βάρος του κάθε ερωτήματος στο συνολικό βαθμό.

Το διαγώνισμα γίνεται με **κλειστό βιβλίο** και **κλειστές σημειώσεις**. Δεν επιτρέπεται να ανατρέξετε σε βιβλίο, σημειώσεις ή άλλο υλικό κατά τη διάρκεια της εξέτασης. Δεν επιτρέπεται επίσης η χρήση αριθμομηχανών, τηλεφώνων και άλλων ηλεκτρονικών συσκευών.

Γράψτε τις απαντήσεις σας πάνω στο χαρτί με τα θέματα. Στις ερωτήσεις όπου χρειάζεται να γράψετε κώδικα, δεν πρέπει να χρησιμοποιήσετε οποιαδήποτε συνάρτηση βιβλιοθήκης, εκτός αν αναφέρεται ρητά στο ερώτημα.

Μόλις τελειώσετε, παραδώστε το συμπληρωμένο διαγώνισμα και το αποδεικτικό ταυτότητας στους διδάσκοντες ή στους επιτηρητές στην έδρα. Αν φεύγετε πριν το τέλος της εξέτασης, παρακαλείστε να μην κάνετε θόρυβο για να μην ενοχλείτε τους συμφοιτητές σας που γράφουν.

Ερώτημα	Βαθμός
1	
2	
3	
Σύνολο	

Μη γράψετε στον παραπάνω πίνακα.

Ονοματεπώνυμο:

A.M.: _____

1 Ερωτήσεις Πολλαπλής Επιλογής (70%)

Επιλέξτε μια από τις προτεινόμενες απαντήσεις σε κάθε ερώτηση, και κυκλώστε τον αριθμό στα αριστερά της απάντησης. Επιλέξτε μια απάντηση για κάθε ερώτηση.

1.1 Array Placement

Υποθέστε τον ακόλουθο πίνακα 2 διαστάσεων:

char A[100][100];

Δίνεται ότι `sizeof(char) == 1` και ότι ο πίνακας ξεκινά στη διεύθυνση 100, δηλαδή `&A == 100`. Σε ποιά διεύθυνση είναι αποθηκευμένο το στοιχείο `A[40][50]` του πίνακα;

1. 4140
2. 5140
3. 4150
4. 5150

1.2 Τί θα τυπώσει το παρακάτω πρόγραμμα;

Υποθέστε ότι ο πίνακας αγγεκινά στη διεύθυνση 2000 και ότι το μέγεθος ενός int είναι 32 bit. Η συνάρτηση printf με το αλφαριθμητικό μορφοποίησης "%u" τυπώνει unsigned int αριθμούς στο δεκαδικό σύστημα αριθμησης.

```
#include <stdio.h>

int main() {
    int arr[5];
    /* Assume that base address of arr is 2000 and size of integer
     * is 32 bit */
    arr++;
    printf("%u", arr);

    return 0;
}
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. 2000
4. 2002
5. 2004
6. 2016

1.3 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>

int main() {
    int i;
    for(i = 0; i < 5; i++) {
        int i = 10;
        printf("%d", i);
        i++;
    }
    return 0;
}
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. 10 11 12 13 14
4. 0 1 2 3 4

5. 1 2 3 4 5
6. 10 10 10 10 10
7. 10

1.4 Τί κάνει η παρακάτω συνάρτηση;

```
void fun(int num) {
    if (num / 8) fun(num / 8);
    putchar(num % 8 + '0');
}
```

1. Μετρά τον αριθμό bits που είναι 1 στην παράμετρο num
2. Τυπώνει την παράμετρο num στο δυαδικό σύστημα
3. Τυπώνει την παράμετρο num στο οκταδικό σύστημα
4. Αθροίζει όλα τα bits στην παράμετρο num
5. Τυπώνει το υπόλοιπο της διαίρεσης της παραμέτρου num με το 8

1.5 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>

int main() {
    float pi = 3.14159f;
    int i;

    i = (int) pi;
    printf("%d", i);
    return 0;
}
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. 3
4. 3.14159
5. 3.000000
6. 3.141590

1.6 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>
struct foo {
    int x;
    static int y;
};

int main()
{
    printf("%d", sizeof(struct foo));
    return 0;
}
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. 4
4. 8

1.7 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>

int main() {
    int i;
    int arr[5] = {0};
    for (i = 0; i <= 5; i++)
        printf("%d ", arr[i]);
    return 0;
}
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα τυπώσει 5 μηδενικά
3. Το πρόγραμμα θα τυπώσει 5 μηδενικά και έναν τυχαίο αριθμό (σκουπίδια).
4. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (segmentation fault).
5. Το πρόγραμμα ή θα τυπώσει 5 μηδενικά και έναν τυχαίο αριθμό ή θα έχει λάθος εκτέλεσης (segmentation fault).
6. Το πρόγραμμα θα τυπώσει 5 μηδενικά και έναν τυχαίο αριθμό και μετά θα τερματίσει με λάθος εκτέλεσης (segmentation fault).

1.8 Debugging

Δίνεται το παρακάτω πρόγραμμα. Η συνάρτηση concat ενώνει δύο αλφαριθμητικά, αντιγράφοντας έναν έναν τους χαρακτήρες του δεύτερου στο τέλος του πρώτου. Επομένως, η αναφορά μενόμενη έξοδος του προγράμματος είναι "Hello world!". Το πρόγραμμα μεταγλωττίζεται χωρίς λάθη μεταγλωττισης, αλλά τερματίζει με segmentation fault όταν εκτελείται.

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

void concat(char *a, char *b) {
    int i, m, n;

    m = strlen(a);
    n = strlen(b);
    for (i = 0; i <= n; i++) {
        a[m+i] = b[i];
    }
}

int main() {
    char *str1 = "Hello ";
    char *str2 = "world!";
    concat(str1, str2);
    printf("%s ", str1);
    return 0;
}
```

Ποιά από τις παρακάτω αλλαγές πρέπει να γίνει ώστε το πρόγραμμα να εκτελεστεί επιτυχώς;

1. Η γραμμή 15 της συνάρτησης main να αλλάξει σε
char str1[100] = "Hello ";
2. Να προστεθεί στο τέλος της συνάρτησης concat η γραμμή
a[m+n-1] = '\0';
3. Να προστεθεί στην αρχή της συνάρτησης concat η γραμμή
a = (char *)malloc(sizeof(char)*(strlen(a) + strlen(b) + 1))

1.9 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>
void swap(char *str1, char *str2) {
    char *temp = str1;
    str1 = str2;
    str2 = temp;
}
int main() {
    char *str1 = "bar";
    char *str2 = "foo";
    swap(str1, str2);
    printf("%s %s", str1, str2);
    return 0;
}
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. bar foo
4. foo bar
5. foo foo
6. bar bar

1.10 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>
int main() {
    char str[20] = "csd 2025";
    printf ("%d", (int) sizeof(str));
    return 0;
}
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. 8
4. 9
5. 20
6. 21

1.11 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>

void f(char *s) {
    if (*s != '\0' && *s != ' ') {
        f(s + 1);
        putchar(*s);
    }
}

int main() {
    f("Ace Of Spades");
    return 0;
}
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. Ace Of Spades
4. Ace
5. ecA
6. sedapS fO ecA

1.12 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
#include <stdio.h>
int main() {
    printf("%ld", (long) main);
    return 0;
}
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. Τη διεύθυνση της συνάρτησης main
4. Μια τυχαία τιμή κάπου στη μνήμη (σκουπίδια)

1.13 Τι θα τυπώσει το παρακάτω πρόγραμμα;

```
#include <stdio.h>
int fun(int);
int main()
{
    int (*ptr)(int) = fun;
    (*ptr)(3);
    return 0;
}
int fun(int n)
{
    for(;n > 0; n--)
        printf("12345 ");
    return 0;
}
```

1. Το πρόγραμμα θα έχει λάθος κατά τη μεταγλώττιση
2. Το πρόγραμμα θα έχει λάθος κατά την εκτέλεση (crash)
3. 12345 12345 12345 12345
4. 12345 12345 12345
5. 12345 12345

1.14 Preprocessor

Σε ποιά γραμμή του παρακάτω προγράμματος θα εμφανίσει λάθος μεταγλώττισης ο gcc και γιατί;

```
#include<stdio.h>
#define CIRCUM(R) (3.14*R*R);
int main() {
    float r = 1.0f, c;
    c = CIRCUM(r);
    printf("%f\n", c);
    if(CIRCUM(r) == 6.28)
        printf("DONE\n");
    return 0;
}
```

1. Γραμμή 2, λάθος χρήση παρενθέσεων.
2. Γραμμή 4, συντακτικό λάθος στην αρχικοποίηση.
3. Γραμμή 5, η μακρο-εντολή CIRCUM περιμένει R αντί r.

4. Γραμμή 5, η μακρο-εντολή CIRCUM θα προκαλέσει συντακτικό λάθος.
5. Γραμμή 7, η μακρο-εντολή CIRCUM θα προκαλέσει συντακτικό λάθος.
6. Γραμμή 8, λείπουν τα άγκιστρα από το μπλοκ της εντολής if.

2 Λίστες (30%)

Δίνεται ο τύπος ενός στοιχείου της λίστας:

```
struct list_node {  
    int data;  
    struct list_node *next;  
};
```

Να γράψετε ένα πρόγραμμα που θα δημιουργεί την παρακάτω λίστα:



3 Δείκτες σε συναρτήσεις (20%)

3.1 Δήλωση

Θεωρήστε τις συναρτήσεις:

```
float add(float a, float b);
float subtract(float a, float b);
float multiply(float a, float b);
float divide(float a, float b);
```

Να δηλώσετε ένα καθολικό πίνακα δεικτών F, και να τον αρχικοποιήσετε με δείκτες σε συναρτήσεις αυτές.

3.2 Χρήση

Δίνεται ακόμη μια συνάρτηση που χρησιμοποιεί τις παραπάνω:

```
float switch_function(char op, float a, float b) {
    float result;
    switch(op) {
        case 0:
            result = add(a, b); break;
        case 1:
            result = subtract(a, b); break;
        case 2:
            result = multiply(a, b); break;
        case 3:
            result = divide(a, b); break;
    }
    return result;
}
```

Χρησιμοποιώντας τον πίνακα του προηγούμενου υποερωτήματος, να ξαναγράψετε συνάρτηση χωρίς εντολές απόφασης (if, switch, κλπ).

ΗΥ-100: Εισαγωγή στην Επιστήμη Υπολογιστών **Επαναληπτικό Διαγώνισμα**

Οδηγίες

Διαβάστε προσεκτικά ολόκληρο το διαγώνισμα και σκεφτείτε το χρόνο που θα αφιερώσετε στο κάθε ερώτημα. Λάβετε υπόψη το σχετικό βάρος του κάθε ερωτήματος στο συνολικό βαθμό.

Το διαγώνισμα γίνεται με **κλειστό βιβλίο** και **κλειστές σημειώσεις**. Δεν επιτρέπεται να ανατρέξετε σε βιβλίο, σημειώσεις ή άλλο υλικό κατά τη διάρκεια της εξέτασης. Δεν επιτρέπεται επίσης η χρήση αριθμομηχανών, τηλεφώνων και άλλων ηλεκτρονικών συσκευών.

Γράψτε τις απαντήσεις σας πάνω στο χαρτί με τα θέματα. Στις ερωτήσεις όπου χρειάζεται να γράψετε κώδικα, δεν πρέπει να χρησιμοποιήσετε οποιαδήποτε συνάρτηση βιβλιοθήκης, εκτός αν αναφέρεται ρητά στο ερώτημα.

Ονοματεπώνυμο

A.M.:

1 Multiple Choice (40%)

Επιλέξτε μια από τις προτεινόμενες απαντήσεις σε κάθε ερώτηση, και κυκλώστε τον αριθμό στα αριστερά της απάντησης.

1.1 Ποιός είναι ο σωστός τρόπος να δηλώσουμε ένα δείκτη σε συνάρτηση, η οποία δέχεται μια ακέραια παράμετρο και επιστρέφει έναν ακέραιο;

1. *(int ptrFunc(int));
2. int ()(int)* ptrFunc;
3. int *ptrFunc(int);
4. int (*ptrFunc)(int);
5. (int *ptrFunc)(int);

1.2 Πόσες Θέσεις (bytes) μνήμης δεσμεύει η αλφαριθμητική σταθερά "Information";

1. 11
2. 12
3. 13
4. 14

1.3 Οι παρακάτω δύο εντολές είναι ισοδύναμες (Θα παράξουν τις ίδιες τιμές για τα a, b, x);

```
x = ++a + b++;
x = a++ + ++b;
```

1. Ναι
2. Όχι

1.4 Ποιά θα είναι η τιμή της μεταβλητής x στο τέλος του παρακάτω προγράμματος;

```
int main() {
    int x = 20;
    for(x = 0; x < 10; ++x) {
    }
    /* What is the value of x here? */
}
```

1. 0
2. 9
3. 10
4. 19
5. 20
6. 11

1.5 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
int main() {
    int a[] = {10, 20, 30};
    printf("%d", *a + 1);
}
```

1. 10
2. 20
3. 30
4. 11
5. 21
6. 31

1.6 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
int main() {
    int i = 0;
    do {
        int i = 10;
        printf("%d", i);
        i++;
    } while (i++ < 5);
    return 0;
}
```

1. 10 11 12 13 14
2. 0 1 2 3 4 5
3. 0 1 2 3 4
4. 1 2 3 4 5
5. 10 10 10 10 10 10
6. 10

1.7 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>

int main() {
    float x = 3.14159f;
    int i;

    i = (int) x;
    printf("%d", i);
    return 0;
}
```

1. 3
2. 3.1
3. 3.000000
4. 3.141590

1.8 Τί θα τυπώσει το παρακάτω πρόγραμμα;

```
#include<stdio.h>
int main() {
    int i = 5, *j, **k;
    j = &i;
    k = &j;
    printf("%d %d %d", *j, **k, *(*k));
    return 0;
}
```

1. 5 5 5
2. 0 0 0
3. 6 6 6
4. 5 5 0

2 Code Understanding (20%)

2.1 Εξηγήστε τι λειτουργία κάνει το παρακάτω πρόγραμμα.

```
#include<stdio.h>
int main(int argc, char** argv) {
    while(argc--) {
        printf("%s\n", argv[argc]);
    }
    return 0;
}
```

2.2 Εξηγήστε τι λειτουργία κάνει το παρακάτω πρόγραμμα.

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

void f(char s[]) {
    int i, j;
    char c;

    for (i = 0, j = strlen(s) - 1; i < j; i++, j--) {
        c = s[i];
        s[i] = s[j];
        s[j] = c;
    }
}

int main() {
    char s[] = "Hello world!";
    f(s);
    printf("Output: %s", s);
    return 0;
}
```

3 Development (40%)

Δηλώστε ένα τύπο struct που να περιγράφει εγγραφές της μορφής struct movie):

- Ένας ακέραιος (int) που περιέχει τον αριθμό χαρακτήρων του ονόματος (μαζί με το '\0').
- Το όνομα της ταινίας με μήκος τόσους χαρακτήρες όσο ο προηγούμενος ακέραιος αριθμός.
- Ένας ακέραιος (int) που περιέχει τη χρονιά παραγωγής της ταινίας.
- Ένας αριθμός κινητής υποδιαστολής (float) που περιέχει τον μέχρι τώρα τζίρο της ταινίας.

Γράψτε μια συνάρτηση read_movie η οποία θα δέχεται ως όρισμα ένα FILE * και θα διαβάζει από το αντίστοιχο αρχείο, στο σημείο που βρίσκεται η κεφαλή ανάγνωσης, μια δομή struct movie, την οποία θα αποθηκεύει στη μνήμη δεσμεύοντας κατάλληλο χώρο με την συνάρτηση malloc(size_t), και θα επιστρέφει ένα δείκτη στη διεύθυνση της δομής. Μπορείτε να χρησιμοποιήσετε τη συνάρτηση fread(void *, size_t, size_t, FILE *).

4 Out of the box (5%, bonus)

Να γράψετε ένα πρόγραμμα C που να τυπώνει το μήνυμα “hello world” χωρίς να περιέχει κανένα χαρακτήρα semicolon “;”. Μπορείτε να χρησιμοποιήσετε τη συνάρτηση printf από τη βιβλιοθήκη stdio.h.

HY-100

Τελικό διαγώνισμα 2012-2013
Δευτέρα 28 Ιανουαρίου 2013

Ασκηση 1^η (25%)

Γραψτε τι κανει η παρακατω συναρτηση. Εξηγηστε

```
void f(char* s, char* t){  
    while(*s++=*t++)!='\0');  
}
```

Ασκηση 2^η (25%)

Τι θα εμφανισει το παρακατω προγραμμα;

```
int main()  
{  
    char c;  
    for(c='a';c<'g';++c)  
    {  
        switch (c)  
        {  
            case('a'): c+=2;  
            case('c'): c+=1;  
            case('d'): c++;  
            printf("%c\n",c--);  
            default:c++;  
        }  
    }  
    printf("****%c",c);  
    return 0;  
}
```

Ασκηση 3^η (25%)

- α) Φτιαξτε μια δομη point που θα περιεχει τις συντεταγμενες x και y σε 2 δεκαδικους διπλης ακριβειας με το ίδιο ονομα(x και y)
- β) Φτιαξτε μια συναρτηση insert_point που θα παιρνει ως παραμετρους 2 δεκαδικους διπλης ακριβειας,θα δημιουργει ένα δεικτη σε point θα τον αρχικοποιει με τους δοσμενους αριθμους και θα τον επιστρεφει

Ασκηση 4^η (25%)

Φτιαξτε μια συναρτηση min_arr η οποια θα δεχεται έναν πινακα ακεραιων και το μεγεθος του. Θα υπολογιζει το μικροτερο στοιχειο και θα το επιστρεφει.

Ασκηση 5^η (25%)

Το παρακατω προγραμμα υπολογιζει το παραγωντικο του 5. Όμως το αποτελεσμα που εμφανιζει είναι 24 ενώ το παραγωντικο του 5 είναι 120. Γιατι συμβαινει αυτό; Διορθωστε το λαθος.

```
int main()
{
    int total=1;
    int number=5;
    while(number>1)
    {
        total*=--number;
    }
    printf("%d\n",total);
    return 0;
}
```

ΗΥ-100: Εισαγωγή στην Επιστήμη Υπολογιστών

Οδηγίες

Διαβάστε προσεκτικά ολόκληρο το διαγώνισμα και σκεφτείτε το χρόνο που θα αφιερώσετε στο κάθε ερώτημα. Λάβετε υπόψη το σχετικό βάρος του κάθε ερωτήματος στο συνολικό βαθμό.

Το διαγώνισμα γίνεται με **κλειστό βιβλίο** και **κλειστές σημειώσεις**. Δεν επιτρέπεται να ανατρέξετε σε βιβλίο, σημειώσεις ή άλλο υλικό κατά τη διάρκεια της εξέτασης. Δεν επιτρέπεται επίσης η χρήση αριθμομηχανών, τηλεφώνων και άλλων ηλεκτρονικών συσκευών.

Στις ερωτήσεις όπου χρειάζεται να γράψετε κώδικα, δεν πρέπει να χρησιμοποιήσετε οποιαδήποτε συνάρτηση βιβλιοθήκης, εκτός αν αναφέρεται ρητά στο ερώτημα.

Η εξέταση διαρκεί 2 ώρες.

1 Ανάγνωση κώδικα (20%)

Τί είδους λειτουργία υλοποιεί η παρακάτω συνάρτηση; Εξηγήστε. Προτείνετε ένα καλύτερο όνομα.

```
void f(char *s, char *t) {  
    while((*s++ = *t++) != '\0') ;  
}
```

2 Δομές (20%)

- Δηλώστε ένα τύπο δομής `point` που αποθηκεύει τις συντεταγμένες x και y ενός σημείου ως αριθμούς κινητής υποδιαστολής διπλής ακρίβειας σε πεδία με αντίστοιχο όνομα (x και y).
- Γράψτε μια συνάρτηση `create_point` η οποία να παίρνει δυο αριθμούς κινητής υποδιαστολής διπλής ακρίβειας ως παραμέτρους, να δεσμεύει μνήμη για μια δομή `point`, να αρχικοποιεί τα πεδία της δομής με τις παραμέτρους, και να επιστρέφει ένα δείκτη στη δομή.

3 Αναδρομή (20%)

Το παρακάτω επαναληπτικό πρόγραμμα υπολογίζει το μέγιστο κοινό διαιρέτη:

```
unsigned int gcd_iterative(unsigned int x, unsigned int y) {  
    while(x != y) {  
        if(x < y) y = y - x;  
        else x = x - y;  
    }  
    return x;  
}
```

Να γράψετε μια αναδρομική συνάρτηση η οποία θα υπολογίζει το μέγιστο κοινό διαιρέτη των x και y με τον ίδιο τρόπο, με τύπο:

```
unsigned int gcd(unsigned int x, unsigned int y);
```

4 Πίνακες (20%)

Να γράψετε μια συνάρτηση `arr_avg` που να δέχεται ένα πίνακα ακεραίων και έναν ακέραιο με το μέγεθος του πίνακα, να υπολογίζει το μέσο όρο των στοιχείων του πίνακα, και να το επιστρέψει.

5 Debugging (20%)

Δίνεται το παρακάτω πρόγραμμα. Η συνάρτηση `concat` ενώνει δύο αλφαριθμητικά, αντιγράφοντας έναν έναν τους χαρακτήρες του δεύτερου στο τέλος του πρώτου. Επομένως, η αναμενόμενη έξοδος του προγράμματος είναι "Hello world!". Το πρόγραμμα μεταγλωτίζεται χωρίς λάθη μεταγλώττισης, αλλά τερματίζει με segmentation fault όταν εκτελείται.

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

void concat(char *a, char *b) {
    int i, m, n;

    m = strlen(a);
    n = strlen(b);
    for (i = 0; i <= n; i++) {
        a[m+i] = b[i];
    }
}

int main() {
    char *str1 = "Hello ";
    char *str2 = "world!";
    concat(str1, str2);
    printf("%s ", str1);
    return 0;
}
```

Ποιό είναι το λάθος; Πώς διορθώνεται το πρόγραμμα έτσι ώστε να έχει την αναμενόμενη λειτουργία;

6 Δείκτες σε συναρτήσεις (bonus, 20%)

Ο βαθμός σε αυτό το ερώτημα θα μετρήσει αν στα προηγούμενα ερωτήματα συγκεντρώθηκε άνω του 50%.

6.1 Δήλωση

Θεωρήστε τις συναρτήσεις:

```
float add(float a, float b);
float subtract(float a, float b);
float multiply(float a, float b);
float divide(float a, float b);
```

Να δηλώσετε ένα καθολικό πίνακα `F` του οποίου κάθε στοιχείο είναι ένας δείκτης σε συνάρτηση, και να τον αρχικοποιήσετε με δείκτες στις συναρτήσεις αυτές.

6.2 Χρήση

Δίνεται ακόμη μια συνάρτηση που χρησιμοποιεί τις παραπάνω:

```
float switch_function(char op, float a, float b) {  
    float result;  
    switch(op) {  
        case 0:  
            result = add(a, b); break;  
        case 1:  
            result = subtract(a, b); break;  
        case 2:  
            result = multiply(a, b); break;  
        case 3:  
            result = divide(a, b); break;  
    }  
    return result;  
}
```

Χρησιμοποιώντας τον πίνακα του προηγούμενου υποερωτήματος, να ξαναγράψετε αυτή τη συνάρτηση χωρίς εντολές απόφασης (if, switch, κλπ).

1)

Η συνάρτηση αντιγράφει το string t στο string s. Για κάθε ένα χαρακτήρα του t, μαζί με το \0, το βάζει στο s. Κατόπιν αυξάνονται οι pointer κατά μια θέση. Ένα καλύτερο όνομα θα ήταν copy.

2)

```
struct point {  
    double x,y;  
};  
  
struct point *create_point(double x, double y) {  
    struct point *new_point = malloc(sizeof(struct point));  
  
    new_point->x = x;  
    new_point->y = y;  
  
    return new_point;  
}
```

3)

```
unsigned gcd(unsigned x, unsigned y) {  
    if (x == y) return x;  
  
    if (x < y) y = y - x;  
    else x = x - y;  
  
    return gcd(x, y);  
}
```

4)

```
double arr_avg(int *arr, int n) {  
    int sum = 0;  
  
    for (int i = 0; i < n; i++) {  
        sum += arr[i];  
    }  
  
    // diaforetika 8a ginei akeraia diairesi  
    return (double)sum / n;  
}
```

5)

str1 και str2 είναι read-only strings, δεν μπορούμε να τα αλλάξουμε.

Η λύση θα ήταν να δηλώσουμε το str1 ως str1[50], το 50 είναι ενδεικτικό δεν έκατσα να μετρήσω ακριβώς πόσο θέλει.

6)

i) `float (*funcs[])(float, float) = {add, subtract, multiply, divide};`

ii)

```
float switch_function(char op, float a, float b) {
    return funcs[op](a, b);
}
```