

סמסטר א' תשס"ט, מועד א'

8/2/2009

משך הבחינה: שעתיים

חומר עזר: שני דפי עזר

בחינה בקורס: פרויקט תוכנה
מרצים: ד"ר רוזד שרן וגב' מיכל עוזרי-פלטו

הנחיות כלליות לבחינה:

- בראש העמוד הראשון של טופס המבחן (עמוד זה) יש לציין את מספר תעודת הזהות.
- בבחינה שש שאלות (פתוחות ואמריקאיות) בעלות ניקוד משתנה, בסך של 105 נק'.
- את התשובות לשאלות האמריקאיות יש למלא בטבלה המיועדת לכך (בעמוד זה).
- חובה לתעד את התשובות לשאלה הפתוחה (כמובן מותר בעברית).
- בתשובה לשאלה הפתוחה יש לכלול את כל ההכרזות הדרושות, אך אין צורך להוסיף לקטעי הקוד פקודות #include.

בהצלחה !

טבלת תשובות לשאלות האמריקאיות

✓	ב	שאלה 2
✓	ב	שאלה 3
✓	3	שאלה 4
✓	2	שאלה 5
✓	ב	שאלה 6

שאלה מס' 1 (65 נק')

כתבו תוכנית המקבלת שני ארגומנטים: שם קובץ קלט ושם קובץ פלט. בקובץ הקלט מאוחסנים בפורמט טקסט מספרים שלמים חיוביים (int) שמספרם לא ידוע מראש, מופרדים על ידי רווחים. על התוכנית לקרוא את המספרים הללו, למייןם בעזרת פונקציה הספריה qsort ולהדפיסם ממויינים לקובץ הפלט (שוב בפורמט טקסט כאשר המספרים מופרדים על ידי רווחים). אבל, שימו לב: המיון עצמו יתבצע לפי סכום הספרות של כל מספר ולא לפי ערכו המספרי. לצורך כך, כחלק מהפתרון, עליכם לממש פונקציה המקבלת מספר שלם חיובי (int) ומחזירה את סכום ספרותיו.

שאלה מס' 2 (10 נק')

מהו הפלט של קטע הקוד הבא משמאל לימין (הניחו קיום כל ההכרזות הדרושות):

```
void f2(int *pn, int *pm){
    pn = pm;
    *pm = *pm ? 1:-1;
}
int main(){
    int n=2, m=0;
    int *pn = &n, *pm = &m;
    f2(pn, pm);
    printf("%d %d", *pn, *pm);
    return 0;
}
```

- א. -1 -1
- ב. 1 1
- ג. 2 -1
- ד. 2 1

שאלה מס' 3 (10 נק')

מהו הפלט של קטע הקוד הבא משמאל לימין (הניחו קיום כל ההכרזות הדרושות):

```
int main(){
    int i, n=6;
    for (i=0; i < 3; i++){
        printf("%d ",(n>>i)&1);
    }
    return 0;
}
```

- א. 000
- ב. 011
- ג. 100
- ד. 111

שאלה מס' 4 (10 נק')

מהו הפלט של קטע הקוד הבא משמאל לימין (הניחו קיום כל ההכרזות הדרושות):

```
int f1(int n) {
    n %=3;
    return n;
}
int main(){
    int n=5;
    int m = f1(n);
    printf("%d %d",n, m);
    return 0;
}
```

- א. 1 1
- ב. 2 2
- ג. 5 1
- ד. 5 2

שאלה מס' 5 (5 נק')

נתונה התוכנית הבאה (הניחו קיום כל ההכרזות הדרושות):

```
int main(){
    short a[3][4];
    char *pc1, *pc2;
    pc1 = (char*)(a+1);
    pc2 = (char*)(a+2);
    pc1++;
    pc2--;
    printf("%d ",pc2-pc1);
    return 0;
}
```

מה פלט התוכנית בהנחה ש- $\text{sizeof(char)}=1$ ו- $\text{sizeof(short)}=2$?

- א. 0
- ב. 2
- ג. 4
- ד. 6

שאלה מס' 6 (5 נק')

מהו הפלט של קטע הקוד הבא משמאל לימין (הנח קיום כל ההכרזות הדרושות):

```
int n=2;
int f3(){
    static int m=1;
    n*= m;
    m = n+1;
    return n;
}
int main(){
    int m;
    n = f3();
    m = f3();
    printf(“%d %d”,n, m);
    return 0;
}
```

- א. 22
- ב. 66
- ג. 62
- ד. 26

בהצלחה!

שאלה 1

הנחיה: כפי שצוין במהלך ההבחן, ניתן להשתמש ב-assert, ולפניכן לא יהיו שגיאות ויזעזועים וזיכרון במקומות הנכונים, כיון שה-assert בא לשמור כי לא נהרסו נתונים.

{ include-ה-ב
הנכונים

/* Prototypes: */

int compare_int (const void *p1, const void *p2); /* qsort */

int sumOfDigits (int n); /* למציאת סכום ספרות */

int main (int args_num, char **args) {

/* פתרון ב-2017 */

FILE *in, *out;

int *arr, size = 0, i = 0, tmp;

/* השם גבוהים לקבצים
הקובץ קטן לפי גודל */

assert (args_num >= 3); /* הקובץ מספר אחרים */

in = fopen (args [1], "r");

assert (in != NULL); /* הקובץ קיים והוא input */

out = fopen (args [2], "w");

assert (out != NULL); /* קובץ output */

/* היקבין זה לא אחרת, היינו צריכים להראות in

/* קובץ קטן מספר המספרים ב-in */

while (i = fscanf (in, "%d", &tmp)) {

size ++;

assert (i != EOF); /*

כאן אנחנו חייבים שיש לנו קובץ, בומר, אין לו יותר int קובץ, ויחסים 0 (מספר המספרים המינימלי), וכן במקרה של שגיאה בקובץ, תמונה שגיאה של EOF, ויחסים של i.

/* שגיאה קטנה מאוד, המספר המצויים ומספרה ב-in */

arr = (int *) (malloc (size * sizeof (int)));

assert (arr != NULL); /* הקובץ הוא לא null, לכן אלו מספרים, out, in */

rewind (in); /* כיוון שיש לנו את האררי ויש לנו את האררי */

Handwritten notes and scribbles on the left side of the page, including a circled '2' and various illegible markings.

Handwritten notes at the bottom left: "מקבילי אלו", "מספר, היינו", "בזמן שיש לנו", "in ו-out".

```

/* arr -> array of numbers */
for (i = 0; i < size; i++) {
    tmp = fscanf(in, "%d", &arr[i]);
    assert (tmp != EOF); /* if fscanf returns EOF, it means the input file is empty */
    arr[i] = sumOfDigits(arr[i]);
}

```

```

/* arr -> array of numbers, arr is sorted */
qsort(arr, size, sizeof(int), compare_int);

```

```

/* out -> output file */

```

```

tmp = fprintf(out, "%d", arr[0]);
assert (tmp == 1);
for (i = 1; i < size; i++) {
    tmp = fputc(' ', out);
    assert (tmp != EOF);
    tmp = fprintf(out, "%d", arr[i]);
    assert (tmp == 1);
}

```

```

/* arr is sorted, arr is array of numbers */

```

```

fclose(in);
fclose(out);

```

```

free(arr);
return 0;

```

```

/* $110 */

```

```

/* arr is array of numbers */

```

(B)
 ...
 ...

```

/* qsort size int-f nlls -dpsd */
int compare_int (const void *p1, const void *p2) {
    const int *q1 = p1, *q2 = p2;
    return ((*q1) - (*q2));
}

```

```

/* def -noe-so-son -dpsd */
int sumOfDigits (int n) {

```

```

    int sum = 0;

```

```

    while (n != 0) {

```

```

        sum += n % 10; /* n mod 10 */

```

```

        n = n / 10;

```

```

    }

```

```

    return sum;
}

```