

סמינר א' תשע"ב מועד א
תאריך: 10.2.2012
שעה: 9:00
משך הבחינה: 4 שעות
כל חומר עדר אסור בשימוש



--

בחינה בקורס: מבוא למדעי המחשב

- יש לענות על כל 5 השאלות.
- בכל שאלות בבחן יש לכתוב פונקציות ייעילות ככל האפשר גם בזמן וגם בזכרון.
- שימרו על סגנון התכניות. הגדרו קבועים היכן שצרכיך וכתבו פונקציות עזר.
- יש לשחרר את כל שטחי הזיכרון שהוקטו דינאמית כאשר אין בהם צורך יותר.
- אין צורך לבדוק הצלחה של הקצאות דינמיות.
- בכל שאלות בבחן ניתן להניח שהקלט תקין.
- אלא אם נאמר אחרת, ניתן להשתמש (ambil למש) בכל הפונקציות הנתונות בעמוד 2 של מבחן זה.
- אין צורך לטעד את הפונקציות.

- יש לכתוב ת"ז ומספר מחברת בראש כל עמוד. עמוד ללא ת"ז בראשו לא יבדק.
- יש לענות על כל שאלה במקום שמיועד לה בגוף השאלה. המחברות הן טיווח בלבד ולא תיבדקנה.
- שימו לב כי הבחינה כוללת 16 עמודים, כולל עמוד זה.

בהצלחה!

רשימת פונקציות ספריה מותרונות לשימוש:

פונקציה	ספריה
int strlen(char str[])	String.h
void strcpy(char dest[], char src[])	String.h
void strcat(char dest[], char src[])	String.h
int strcmp(char s1[], char s2[])	String.h
int __max(int a, int b)	stdlib.h
int __min(int a, int b)	stdlib.h
double pow(double base, double exp)	math.h
double sqrt(double x)	math.h
double fabs(double x)	math.h
cout	iostream
cin	iostream
char cin.get(char& outCh)	iostream
void cin.getline(char outStr[], int max, char delimiter)	iostream

שאלה 1 (20 נקודות):

כתבו את הפונקציה

```
char* octalToBin(char octalNum[]);
```

הfonקציה מקבלת כקלט מחרוזת `octalNum`, המכילה ספרות בין '0' ל-'7' המהוות הצגה אוקטלית (בבסיס 8) של מספרשלם וחובי.

הfonקציה מייצרת ומחזירה מחרוזת חדשנה המכילה את הספרות '0' ו-'1', המהוות את הייצוג הבינארי (בבסיס 2) של המספר המיוצג ב-`octalNum`.

הערה: במחזרות המוחזרת ייתכנו **לכל** היותר שני אפסים מובילים.

למשל

```
octalToBin("721") תחזיר את המחרוזת "111010001".
```

```
octalToBin("7") תחזיר את המחרוזת "111".
```

```
octalToBin("12") תחזיר את המחרוזת "001010".
```

```
char* octalToBin(char octalNum[])
{
    int octalLen ;
    char * binNum ;
    char * currPos ;

    octalLen = strlen(octalNum) ;
    binNum = new char [3*octalLen + 1] ;
    currPos = binNum ;
    for (i = 0 ; i < octalLen ; i++)
    {
        smallOctalToBin(octalNum[i],currPos) ;
        currPos += 3 ;
    }
    binNum[octalLen*3] = '\0' ;
    return(binNum) ;
}
```

```
void smallOctalToBin(char ch, char * binarr)
{
    int numvalue ;

    numvalue = (int) (ch - '0') ;
    if ((numvalue / 4) == 1)
        binarr[0] = '1' ;
    else
        binarr[0] = '0' ;

    if (((numvalue %4) / 2) == 1)
        binarr[1] = '1' ;
    else
        binarr[1] = '0' ;

    if ((numvalue %2) == 1)
        binarr[2] = '1' ;
    else
        binarr[2] = '0' ;
}
```

שאלה 2 (20 נקודות):

כתבו את הפונקציה :

```
int find0(int nums[], int n);
```

הfonקציה מקבלת כקלט מערך num של מספרים שלמים, ואת גודלו n (שלם חיובי). ידוע שהמערך num מקיים את הדרישות הבאות :

1. המערך מכיל את **כל** המספרים : $(n-1, \dots, 1, 2, \dots, 0)$.
2. כל המספרים : $(n-1, \dots, 1, 2, \dots, 0)$ ממוקמים אחד ביחס לשני. כלומר, אם $i \leq j \leq n-1$, אז $i < j$ מופיע במערך לפני j .

על הפונקציה להחזיר את אינדקס התא בו מופיע המספר 0 במערך num.

למשל :

- 4. `find0([1, 2, 3, 4, 0, 5, 6, 7], 8)` ○
- 0. `find0([0, 1, 2, 3, 4, 5, 6, 7], 8)` ○
- 1. `find0([1, 0, 2, 3, 4, 5, 6, 7], 8)` ○

```
int find0(int nums[], int n)
{
    bool found0 ;
    int left, right, mid ;
    int indexOf0 ;

    left = 0 ;
    right = n-1 ;
    found0 = false ;

    while (found0 == false)
    {
        mid = (left + right) / 2 ;
        if (nums[mid] == 0)
        {
            found0 = true ;
            indexof0 = mid ;
        }
        else if (nums[mid] == mid)
            right = mid - 1;
        else
            left = mid + 1 ;
    }
    return indexof0;
}
```

שאלה 3 (20 נקודות):

נתונים הקבועים הבאים:

```
const int MAX_NAME_LEN = 100;
const int MAX_NUM_OF_CHILDREN = 20;
```

נתונה ההגדרה הבאה לייצוג משפחה:

```
struct family{
    char name[MAX_NAME_LEN];           // שדה המכיל את שם המשפחה
    int numChildren;                  // שדה המכיל את מספר הילדים
    int ages[MAX_NUM_OF_CHILDREN];    // מערך של גילאי הילדים
};

typedef struct family FAMILY;
```

כתבו את הפונקציה הבאה:

```
void findLargeFamilies(FAMILY fams[], int& size,
                      int threshold);
```

הפונקציה מקבלת מערך `fams` של משפחות ואת גודלו `size` ובנוסף ערך סף חיובי `threshold`. המערך `fams` והפרמטר `size` הם פרמטרי קלט+פלט.

בקלט המערך `fams` מכיל `size` משפחות כלשأن ובפלט הוא יכיל רק את המשפחות בעלות יותר מ- `threshold` ילדים, אפ"י הסדר המקורי שהיה בינוין.

הפונקציה תעדכן את `size` להיות גודלו הלוגי החדש.

```
void findLargeFamilies(FAMILY fams[], int& size,
                      int threshold)
{
    int readInd, writeInd;

    for (readInd = 0, writeInd = 0; readInd < size;
         readInd++)
    {
        if (fams[readInd].numChildren > threshold)
        {
            fams[writeInd] = fams[readInd];
            writeInd++;
        }
    }

    size = writeInd;
}
```

שאלה 4 (20 נקודות):

כתבו את הפונקציה

```
int* splitByKSize(int inData[], int size, int k);
```

הfonקציה מקבלת כקלט מערך `inData` של מספרים שלמים חיוביים, את גודלו `size` (שלם חיובי) וערך שלם חיובי `k` על פיו תבוצע חלוקה.

הfonקציה מייצרת ומוחזירה מערך חדש ובו כל המספרים של `inData` מסודרים לפי החקיקות הבאה:

ראשית, כל מספרי `inData` שהם בתחום: $\{0, 1, \dots, k-1\}$

אחריהם, כל המספרי `inData` שהם בתחום: $\{k, k+1, \dots, 2*k-1\}$

אחריהם, כל המספרי `inData` שהם בתחום: $\{2*k, 2*k+1, \dots, 3*k-1\}$

...

אחריהם, כל המספרי `inData` שהם בתחום: $\{(k-1)*k, (k-1)*k+1, \dots, k*k-1\}$

ולבסוף כל המספרי `inData` שגדולים או שווים ל: $.k*k$

שימוש לב:

- סדר המספרים בתוך כל תחום אינו חשוב.
- ייתכן בחילט שחלק מהתחומיים לא יכללו אף מספר מתוך `.inData`.

דוגמאות

פלט אפשרי לקריאה

```
sortByKSize([13,100,12,24,2,31,5,19,79,20,30],11,5)
```

הוא המערך `[2,5,13,12,19,24,20,100,31,79,30]`

```
int* sortByKSize(int inData[], int size, int k)
{
    int *counts, *writeIndex, *resArr ;
    int currRange, i ;

    counts = new int [k+1] ;
    for (i = 0 ; i <= k ; i++)
        counts[i] = 0 ;
    for (i = 0 ; i < size ; i++)
    {
        currRange = inData[i] / k ;
        if (currRange >= k)
            counts[k] ++ ;
        else
            counts[currRange] ++ ;
    }

    writeIndex = new int [k+1] ;
    writeIndex[0] = 0 ;
    for (i = 1; i <= k ; i++)
        writeIndex[i] = writeIndex[i-1] + counts[i-1] ;
    delete [] counts ;
```

```
resArr = new int [size] ;
for (i = 0 ; i < size ; i++)
{
    currRange = inData[i] / k ;
    if (currRange >= k)
    {
        resArr[ writeIndex[k] ] = inData[i] ;
        writeIndex[k] ++ ;
    }
    else
    {
        resArr[ writeIndex[currRange] ] = inData[i] ;
        writeIndex[currRange] ++ ;
    }
}

delete[] writeIndex ;
return resArr ;

}
```

שאלה 5 א (10 נקודות)

כתבו מימוש **רקורסיבי** של הפונקציה:

```
int sumPositive(int data[], int size);
```

הfonקציה מקבלת כקלט מערך `data` של מספרים שלמים, ואת גודלו `size` (שלם חיובי) ומחזירה את סכום המספרים החסוביים במערך.

דוגמאות:

הקריאה `.(2+5+8) sumPositive([2,-3,5,-7,8,-1],6)` תחזיר 15

הקריאה `.sumPositive([0,-2,4],3)` תחזיר 4

הקריאה `.sumPositive([-2,-3],2)` תחזיר 0.

```
int sumPositive(int data[], int size)
{
    int tempSum ;

    if (size == 1)
    {
        if (data[0] > 0)
            return( data[0] ) ;
        else
            return(0) ;
    }

    else
    {
        tempSum = sumPositive (data, size-1) ;
        if (data[size-1] > 0)
            return( tempSum + data[size-1] ) ;
        else
            return( tempSum ) ;
    }
}
```

שאלה 5ב (10 נקודות):

כתבו מימוש רקורסיבי של הפונקציה

```
bool subsetMul(int numbers[], int size, int mul);
```

הפונקציה מקבלת כקלט מערך `numbers` של מספרים שלמים, ואת גודלו `size` (שלם חיובי) ומספר נוסף, `mul`.

הפונקציה תחזיר true אם ורק אם קיימת איזושהי תת-קובוצה של מספרים מתוך המערך (לאו דווקא רצופה) שמכפלת המספרים בה שווה ל- `mul`.

דוגמאות

הקריאה `subsetMul([17, -3, 2, -1, 6], 5, -18)`, כי קיימת תת-קובוצה $\{-3, 6\}$.

הקריאה `subsetMul([17, -3, 2, -1, 6], 5, 17)`, כי קיימת תת-קובוצה (শ্মכילה איבר בודד) $\{17\}$.

הקריאה `subsetMul([17, -3, 2, -1, 6], 5, 23)`, כי לא קיימת תת-קובוצה מתאימה.

```

bool subsetMul(int numbers[], int size, int mul)
{
    bool tempres ;
    if (size == 1)
    {
        if (mul == numbers[0])
            return ( true ) ;
        else
            return ( false ) ;
    }
    else
    {
        tempres = subsetMul(numbers, size-1, mul) ;
        if (tempres == true)
            return(true) ;
        else
        {
            if (numbers[size-1] == 0)
            {
                if (mul == 0)
                    return ( true ) ;
                else
                    return( false ) ;
            }
            else
            {
                if(mul % numbers[size-1]== 0)
                    return (subsetMul (numbers, size - 1,
                           mul/ numbers[size-1] ) ;
                else
                    return false;
            }
        }
    }
}

```