מבוא למדעי מחשב מ' / ח' (234114 / 234117)

סמסטר אביב תשס"ה

פתרון מבחן מסכם מועד א', 5 יולי 2005

מספר סטודנט	*שם משפחה*	*שם פרטי*

משך המבחן: 2.5 שעות.
חומר עזר: אין להשתמש בכל חומר עזר בכתב, מודפס או אלקטרוני.

הנחיות והוראות:

- מלאו את הפרטים בראש דף זה.
- בדקו שיש 12 עמודים (5 שאלות) במבחן, כולל עמוד זה.
- כתבו את התשובות על טופס המבחן בלבד, במקומות המיועדים לכך. שימו לב שהמקום המיועד לתשובה אינו מעיד בהכרח על אורך התשובה הנכונה.
- העמודים הזוגיים בבחינה ריקים. ניתן להשתמש בהם כדפי טיוטה וכן לכתיבת תשובותיכם. סמנו טיוטות באופן ברור על מנת שהן לא תבדקנה.
- יש לכתוב באופן ברור, נקי ומסודר. ניתן בהחלט להשתמש בעיפרון ומחק.
- אין לכתוב הערות והסברים לתשובות אם לא נתבקשתם מפורשות לכך.
- בכל השאלות, הינכם רשאים להגדיר (ולממש) פונקציות עזר כרצונכם.
- **אין להשתמש בפונקציות ספריה או בפונקציות שמומשו בכיתה אלא אם צויין אחרת בשאלה.**

שאלה	ערך	הישג	בודק
1	20		
2	20		
3	20		
4	20		
5	20		
סה"כ	100		

צוות הקורס 234114

מרצים: איתן אביאור, דר' רועי פרידמן (מרצה אחראי).

מתרגלים: רן רובינשטיין, מיכל הולצמן-גזית.

צוות הקורס 234117

מרצים: רועי מלמד, דר' רועי פרידמן (מרצה אחראי).

מתרגלים: עידו פלדמן, גיא פליישר, אייל רוזנברג, אולג רוכלנקו.

בהצלחה!

שאלה 1 (20 נקודות)

סעיף א (8 נקודות)

נתון האתחול הבא של משתנים בתוכנית C:

```
char *arr[]={"Programming", "using", "C language", "is fun!"};
char *p = arr[3];
char *q = *(arr + 1);
*(q+2) = '\0';
```

כתבו מה יודפס ע"י כל אחת מן הפקודות הבאות לאחר ביצוע האתחול הנ"ל:

`printf("%s", arr[2]);`	`C language`
`printf("%s", arr[2]+2);`	`language`
`printf("%s", q);`	`us`
`printf("%c", arr[2][0]);`	`C`
`printf("%s", p+3);`	`fun!`

סעיף ב (12 נקודות)

כתבו את סיבוכיות הזמן והמקום של הקריאות **strange(n,1)** ו-**weird(n,1)** כפונקציה של **n** (כאשר n≥0 טבעי):

.1

```
void strange(int n, int k)
{
  int i;

  if (k > n)
    return;

  for (i = k; i < n; i++)
    printf("?");

  strange(n, k+2);
  return;
}
```

סיבוכיות זמן: Θ(n^2) סיבוכיות מקום: Θ(**n**)

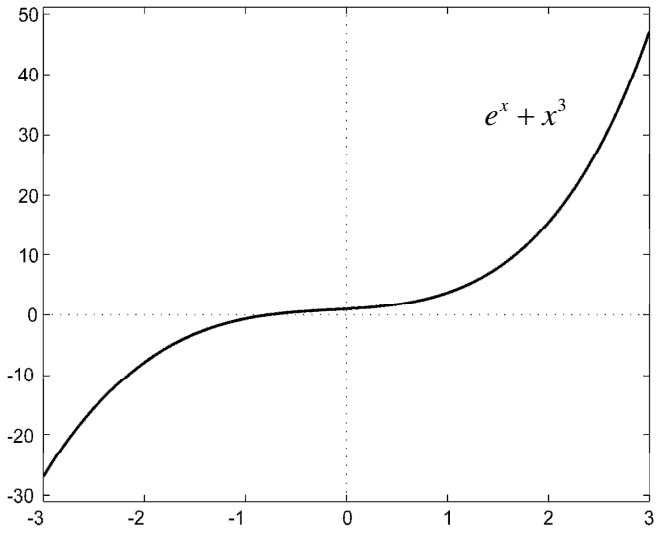
.2

```
void weird(int n, int k)
{
  int i;

  if ((n <= 0) || (k <= 0))
    return;

  k *= 2;

  for (i = 0; i < k; i++) {
    printf("?");
    weird(n/2, -n/2);
  }

  weird(n/2, k);

  return;
}
```

סיבוכיות זמן: Θ(**n**) סיבוכיות מקום: Θ(**log(n)**)

שאלה 2 (20 נקודות)

בשאלה זו נכתוב תוכנית המוצאת את נקודת האפס של הפונקציה $f(x) = e^x + x^3$ (ישנה נקודה יחידה כזו).
להלן גרף המתאר פונקציה זו באופן סכימטי:



סעיף א (10 נקודות)

בסעיף זה נכתוב פונקציה המחשבת את הפונקציה e^x. לשם כך, ידוע שלכל x מתקיים:

$$e^x = 1 + \frac{x}{1} + \frac{x^2}{1\cdot 2} + \frac{x^3}{1\cdot 2\cdot 3} + \frac{x^4}{1\cdot 2\cdot 3\cdot 4} + \cdots$$

כאשר מובטח שהאיברים בזנב הטור הולכים וקטנים לאפס. השלימו את הפונקציה בעמוד הבא, המקבלת מספר x ומחשבת קירוב ל-e^x; על הפונקציה לעצור כאשר התיקון שמתווסף לקירוב הנוכחי קטן בערכו המוחלט מ-**EPSILON** (מוגדר כ-**#define**).

שימו לב: ניתן להעזר בפונקציה **fabs()** שמחשבת ערך מוחלט של מספר ממשי, וחתימתה היא

```
double fabs(double x);
```

אין להעזר בכל פונקציה אחרת בקוד שלכם (ובפרט לא בפונקציה pow()).

```
#define EPSILON 1e-10

double myexp(double x)
{
  double result = 1;
  double n = 1;
  double delta = 1;


  do {
     delta *= x/n;
     results += delta;
     n++;


  } while (     fabs(delta)                 > EPSILON);

  return result;
}
```

לנוחותכם, הנוסחה מובאת כאן בשנית:

$$e^x = 1 + \frac{x}{1} + \frac{x^2}{1\cdot 2} + \frac{x^3}{1\cdot 2\cdot 3} + \frac{x^4}{1\cdot 2\cdot 3\cdot 4} + \cdots$$

<u>סעיף ב (10 נקודות)</u>

נתונה הפונקציה הבאה המחשבת את $f(x) = e^x + x^3$ בנקודה x, ועושה שימוש בפונקציה מהסעיף הקודם:

```
double f(double x) {
  return myexp(x) + x*x*x;
}
```

כתבו פונקציה שמוצאת את נקודת האפס של $f(x)$ (הנקודה בה ערך הפונקציה הוא 0). פונקציה זו תקבל שתי נקודות קצה xmin ו-xmax, כאשר xmin<xmax, ומתקיים ש- f(xmin)<0 ו- f(xmax)>0; על הפונקציה להחזיר את נקודת האפס של f הנמצאת בין שתי נקודות הקצה (מובטח שיש כזו). הפונקציה נדרשת לעבוד בדיוק **DELTA** (קבוע נוסף המוגדר כ-**#define**), כשהכוונה היא **שהנקודה x המוחזרת צריכה להיות לכל היותר במרחק DELTA מנקודת האפס האמיתית x_0**. (שימו לב שהתנאי על הנקודה המוחזרת x איננו תלוי בערך הפונקציה עצמו בנקודה זו.)

בסעיף זה מותר להשתמש בפונקציה **f()** הנתונה גם אם לא פתרתם את הסעיף הקודם.

```
#define DELTA 1e-8

double findzero(double xmin, double xmax) {
  double xmid, val;


  while(xmax-xmin > DELTA)
  {
    xmid = (xmax+xmin)/2;
    val = f(xmid);
    if (val==0)
      return xmin;
    else if (val>0)
      xmax = xmid;
    else
      xmin = xmid;
  }

  return (xmax+xmin)/2;
}
```

שאלה 3 (20 נקודות)

יהי a מערך של מספרים שלמים. **ערך שכיח** ב-a יוגדר להיות כל ערך במערך שמספר הפעמים שהוא מופיע בו הוא מקסימאלי (דהיינו, לא קיים ערך אחר במערך שמופיע בו מספר רב יותר של פעמים). שימו לב שייתכנו כמה ערכים שכיחים עבור מערך מסויים. דוגמאות:

במערך { 2,8,6,2,7,2 } הערך השכיח הוא 2 (הוא מופיע 3 פעמים במערך).
במערך { 1,8,3,2,1,8 } גם 1 וגם 8 הם ערכים שכיחים (כל אחד מהם מופיע פעמיים במערך).

בשאלה זו נכתוב 2 פונקציות **לא רקורסיביות** המקבלות מערך a בגודל n>0 ומחזירות ערך שכיח כלשהו בו (לא משנה איזה). בכל הסעיפים, המערך a איננו ממויין.

סעיף א (10 נקודות)

השלימו את הפונקציה **common1()** להלן, המחזירה ערך שכיח כלשהו במערך a בסיבוכיות זמן O(nlogn) וסיבוכיות מקום O(n). בסעיף זה ניתן להשתמש בכל פונקציה או אלגוריתם שנלמדו בהרצאות. במידה ובחרתם להשתמש בפונקציות עזר שנלמדו בכיתה, ציינו בקצרה באילו פונקציות בחרתם להשתמש.

1. חתימה: `void mergesort(int a[], int n);`

   תאור: ממיינת את המערך a באמצעות אלגוריתם Merge-Sort.

2. חתימה: _____

   תאור: _____

```
int common1(int a[], int n) {
  int i=0, maxcount=0, maxval;
  mergesort(a,n);
  while (i<n) {
    int j = i+1, count = 1, val = a[i];
    while (j<n && a[j]==val) {
      j++; count++;
    }
    if (count > maxcount) {
      maxcount = count;
      maxval   = val;
    }
    i=j;
  }
  return maxval;
}
```

סעיף ב (10 נקודות)

בסעיף זה ניתן להניח כי כל הערכים במערך a הינם בתחום 0..99. השלימו את הפונקציה **common2()** המחזירה ערך שכיח כלשהו במערך a בסיבוכיות זמן O(n) וסיבוכיות מקום O(1). רמז: ניתן להקצות מערך באורך קבוע של 100 ללא חריגה מדרישת סיבוכיות המקום המצויינת.

```
int common2(int a[], int n) {
  int count[100] = {0};
  int i, maxcount, maxval;

  for (i=0; i<n; ++i) {
    count[a[i]]++;
  }

  maxcount = 0;
  for (i=0; i<100; ++i) {
    if (count[i] > maxcount) {
      maxcount = count[i];
      maxval   = i;
    }
  }
  return maxval;
}
```

שאלה 4 (20 נקודות)

נתונה מחרוזת s אשר מכילה רצף של מילים, כאשר בין כל מילה ומילה ישנו תו רווח יחיד (המילים ברצף יכולות להכיל תווים כלשהם, פרט לתווי רווח או NULL). ברצוננו לשנות את המחרוזת s כך שסדר המילים בה יתהפך, כלומר כך שהמילה האחרונה תהפוך להיות הראשונה, המילה הראשונה תהפוך להיות האחרונה, וכן הלאה. שימו לב שברצוננו **לְשַׁמֵּר את המילים עצמן**, כלומר לשמור על סדר האותיות בכל מילה.

לדוגמה, המחרוזת:

```
"please tell me your name"
```

תהפוך ל:

```
"name your me tell please"
```

השלימו את הפונקציה הבאה, אשר מקבלת מחרוזת s כמתואר ומשנה את המחרוזת כנדרש.

דרישות סיבוכיות: סיבוכיות זמן O(n), סיבוכיות מקום נוסף O(1).
שימו לב שהפונקציה צריכה לשנות את המחרוזת המקורית ולא ליצור מחרוזת חדשה.

```
void changeString (char *s) {
  char *t;
  reverse(s,strlen(s));

  while (*s) {
    int n;
    t = nextword(s,&n);
    reverse(s,n);
    s=t;
  }
}


int strlen(char *s)
{
  int i=0;
  while (s[i]) ++i;
  return i;
}
```

```
void reverse(char *s, int k)
{
  int i=0, j=k-1;

  while (i<j) {
    char tmp = s[i];
    s[i] = s[j];
    s[j] = tmp;
    i++; j--;
  }
}


char* nextword(char *s, int *wordlen)
{
  char *t = s+1;
  *wordlen = 1;

  while (*t!=' ' && *t!=0) {
    t++;
    (*wordlen)++;
  }

  if (*t==' ') t++;
  return t;
}
```

שאלה 5 (20 נקודות)

בשאלה זו עליכם לכתוב פונקציה המקבלת שני מספרים טבעיים n ו-k, ומדפיסה את כל האפשרויות להציג את n כסכום של k מספרים טבעיים (לשאלה זו מספר טבעי הוא שלם ≥ 0). **יש להדפיס כל אפשרות פעם אחת בלבד**, ואין לחזור על אותה האפשרות פעמיים – גם לא בסדר שונה. **על הפתרון להיות רקורסיבי ולעבוד בשיטת ה-backtracking.**

שימו לב: בשאלה זו ניתן להשתמש בפונקצית הספרייה **`malloc()`**.

לדוגמה, עבור n=5 ו-k=3 הפונקציה תדפיס את הפלט הבא:

```
5 0 0
4 1 0
3 2 0
3 1 1
2 2 1
```

עבור n=5 ו-k=2 יודפס הפלט הבא:

```
5 0
4 1
3 2
```

ממשו את הפונקציה **`printsum()`** המבצעת את הפעולה המתוארת (שימו לב שביכולתכם לממש פונקציות עזר על פי הצורך). בפתרונכם, ניתן להשתמש בפונקציה הבאה:

```
void printarray(int a[], int n);
```

המקבלת מערך a ואת גודלו n, ומדפיסה את תוכנו.

```
void printsum(unsigned int n, unsigned int k) {
  unsigned int *a = (unsigned int*)malloc(k*sizeof(int));


  f(n,k,a,0);
  free(a);
}
```

```
void f(unsigned int n, unsigned int k, unsigned int a[],int used)
{
  int i;
  if (k==0) {
    if (n==0) printarray(a,used);
    return;
  }
  for (i = (used>0) ? min(a[used-1],n) : n; i>=0; i--) {
    a[used] = i;
    f(n-i,k-1,a,used+1);
  }
}

unsigned int min(unsigned int a, unsigned int b) {
  return (a<b ? a:b);
}
```