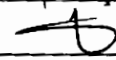


מחברת מס' \_\_\_\_\_  
מתוך \_\_\_\_\_ מחברות

אוניברסיטת תל-אביב  TEL AVIV UNIVERSITY

41

## לפני התחלת הבחינה מלא את כל הפרטים הבאים בכתב ברור וקרא בעיון את ההוראות:

1. על הנבחן להיבחן רק בחדר שבו הוא רשום.
  2. עם הכניסה לחדר הבחינה יש להניח את החפצים בצד לדבות מכשיר קשור ואמצעי תקשורת אחרים כשהם כבויים.
  3. אסור להחזיק בהישג יד, בחדר הבחינה או בסמוך לו, כל חומר הקשור לבחינה/לקורס פרט לחומר שהשימוש בו הותר בכתב על ידי המורה.
  4. יש למלא את הפרטים על מחברת הבחינה במקום המיועד לכך בלבד. אין לכתוב את השם או כל פרט מזהה אחר בתוך המחברת.
  5. יש להישמע להוראות המשיג. נבחן לא יעזוב את מקומו ללא קבלת רשות המשיג. הפונה בשאלה או בבקשה ירים את ידו.
- לשימוש המורה הבוחן:
- |                   |  |
|-------------------|--|
| הציון             | 100  |
| המחברת נבדקה ביום | 27/6/04  |
| חתימת המורה       |  |
6. נבחן שנכנס לחדר הבחינה וקיבל את השאלון (טופס הבחינה) לידו ייחשב כמי שנבחן במועד זה. היה והחליט לא לכתוב את הבחינה, לא יהא רשאי לעזוב את חדר הבחינה, אלא כעבור חצי שעה ממועד תחילתה ולאחר שהחזיר את המחברת והשאלון. ציונו בבחינה יהיה "0".
  7. קריאת השאלון מותרת רק לאחר קבלת רשות המשיג.
  8. יש לכתוב את התשובות בעט, בכתב יד ברור ונקי. נבחן הבוחר לכתוב טיוטה יעשה זאת בעמודו הימני של דפי מחברת הבחינה ויצוין בראש העמוד "טיוטה". אין לתלוש דפים מהמחברת.
  9. מחברות הבחינה שקיבל הנבחן תהיינה בפקוחו ובאחריותו במשך כל הבחינה. בעת יציאה מן החדר יופקדו המחברות והשאלון בידי המשיג.
  10. בתום הבחינה יחזיר הנבחן את המחברות והשאלון ויקבל מידי המשיג את כרטיס הנבחן.
  11. המהג בניגוד להוראות ולימלה סדרי בחינת ודיווח ציונים צפוי להפסקת בחינתו ואף להעמדה לדיון משמעתי.
  12. אין לכתוב מעבר לקו האדום משני צידי הדף.

בהצלחה.

תאריך הבחינה 27.6.04  
שם הקורס תיג 1  
שם המורה שלם הלפרין  
החוג/המנחה אמאליקה

55154

סמסטר ב' תשס"ד  
מועד: א' 27/06/2004  
משך הבחינה: 3 שעות  
חומר עזר: שני דפי עזר

בחינה בקורס: תוכנה 1  
פרופ' דן הלפרין

הנחיות כלליות לבחינה:

- המבחן מורכב משלוש שאלות תכנות ושלוש שאלות "אמריקאיות". יש להשיב על כל השאלות.
- חובה לתעד כל פעולה לא טריוויאלית שנעשית.
- יש לכתוב קוד קריא ויעיל ככל האפשר.
- נא לכתוב בכתב קריא ולא מחובר.

בהצלחה !

ת.ז.: 032790701

טבלת תשובות לחלק האמריקאי

a	b	c	d	e	
✓			✓		שאלה 4 ✓
✓					שאלה 5 ✓
			✓		שאלה 6 ✓

הערות לתשובות בחלק האמריקאי

אמנם רק תשובה נכונה תזכה בניקוד עבור כל שאלה, אולם ניתן לצרף לכל תשובה אמריקאית הסבר קצר. ההסבר לא ייבדק במסגרת הבדיקה הרגילה, אך ניתן יהיה להסתמך עליו במסגרת ערעור, אם יידרש. מומלץ לצרף הסבר לתשובה אמריקאית במיוחד במקרים קיצוניים בהם נראה לך שתשובתך נכונה, אך נראה לך שהיא איננה התשובה שאליה התכוון המרצה. מובן שהסבר שגוי או בלתי סביר לא יועיל בכל מקרה (אך גם לא יזיק, אם ממילא סימנת את התשובה הנכונה).

שאלה 4:

\_\_\_\_\_

שאלה 5:

\_\_\_\_\_

שאלה 6:

\_\_\_\_\_

שאלה מס' 1 (30 נקודות)

הפונקציה הבאה מאחסנת עץ שהשורש שלו מוצבע ע"י head לתוך קובץ בינארי שנפתח בהצלחה לכתיבה.

```
struct node {
    char *      data;
    struct node * left;
    struct node * right;
};
typedef struct node  NODE;

long tree_to_file(NODE *head, FILE *output_file)
{
    int  length, rc;
    long current_offset, left_child_offset, right_child_offset;

    if (head == NULL) return -1;

    current_offset = ftell(output_file);

    left_child_offset = tree_to_file(head->left, output_file);
    right_child_offset = tree_to_file(head->right, output_file);

    length = strlen(head->data);
    rc = fwrite(&length, sizeof(int), 1, output_file);
    assert(rc == 1);
    rc = fwrite(&left_child_offset, sizeof(long), 1, output_file);
    assert(rc == 1);
    rc = fwrite(&right_child_offset, sizeof(long), 1, output_file);
    assert(rc == 1);
    if (length > 0 ) {
        rc = fwrite(head->data, sizeof(char), length, output_file);
        assert(rc == length);
    }
    return current_offset;
}
```

- א. (10 נקודות) הסבר בקצרה ובמדויק כיצד פועלת הפונקציה tree\_to\_file().  
 ב. (20 נקודות) כתוב פונקציה

NODE \* file\_to\_tree(FILE \* input\_file)

הבונה את העץ המתאים מתוך קובץ שנבנה ע"י tree\_to\_file() ומחזירה מצביע לשורש העץ שבנתה.

## שאלה מס' 2 (30 נקודות)

השתמשו במבנה הנתונים הבא:

```
typedef struct image {
    unsigned char * data;      /* the image data */
    unsigned int width;        /* the image width */
    unsigned int height;       /* the image height */
} Image;
```

השדה data מצביע לנתונים המתארים את ערכי הפיקסלים בתמונה, בפורמט זהה לפורמט של תרגיל הבית, כלומר width\*height פיקסלים עוקבים בזיכרון, כל פיקסל מורכב מ-3 רכיבי צבע, כאשר כל רכיב צבע הוא בגודל של byte.

א. (15 נקודות) כתבו פונקציה

```
int insert(unsigned int x, unsigned int y, Image * img, Image * subimg);
```

המקבלת מצביע img לתמונה גדולה, מצביע subimg לתמונה קטנה, ומיקום x,y של פיקסל בתמונה הגדולה img. הפונקציה מחליפה את החלק של התמונה הגדולה, שראשיתו ב-x,y, בתמונה הקטנה. הפונקציה מחזירה 0 במקרה של הצלחה ומספר שלילי במקרה של כשלון. כשלון נובע מאחת מהסיבות הבאות:

- (1) הקלט אינו מתאר תמונה חוקית (אחד השדות מכיל ערך שלילי -1-0).
- (2) התמונה הקטנה הממוקמת ב-x,y חורגת מגבולות התמונה הגדולה.

ב. (15 נקודות) נתונה פונקציה (אין צורך לכתוב אותה!)

```
int scale_image(Image * src, Image * dst);
```

המקבלת מצביע img לתמונה ויוצרת תמונה מוקטנת באופן הבא: הרוחב של תמונת הפלט קטן מהרוחב של תמונת הקלט פי 2 מעוגל כלפי מטה, אלא אם כן התוצאה היא 0. במקרה זה, הרוחב של תמונת הפלט נקבע להיות 1. אותם דברים אמורים לגבי הגובה. הפונקציה מחזירה 0 במקרה של הצלחה ומספר שלילי במקרה של כשלון. שים לב כי הפונקציה מקצה זכרון עבור הנתונים של תמונת הפלט. (בניגוד למימוש של תרגיל הבית, לא מתבצע padding.)

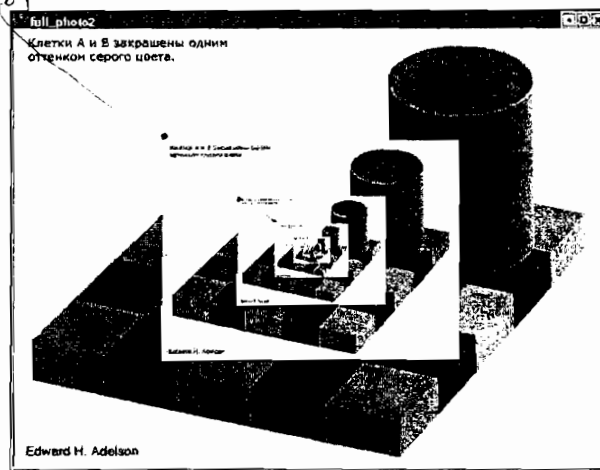
כתבו פונקציה

```
int self_insert(Image *img);
```

המקבלת מצביע img לתמונה ומייצרת סדרה של תמונות בסדר יורד של גודל. כל תמונה בסדרה מתקבלת מהתמונה הקודמת ע"י הפעלת הפונקציה scale\_image(). כל התמונות מאוחסנות בתוך שדה הנתונים של img. כל תמונה בסדרה ממוקמת בתוך תמונת הקלט בעזרת הפונקציה insert() שכתבתם בסעיף הקודם, כך שמרכזי כל התמונות מתלכדים במרכז תמונת הקלט. שימו לב שהרוחב והגובה של התמונה אינם בהכרח זהים ואינם בהכרח חזקה של 2. הפונקציה מחזירה 0 מקרה של הצלחה ומספר שלילי במקרה של כשלון. יש לשחרר את כל הזיכרון שהוקצה ע"י הפונקציה שכתבתם או ע"י scale\_image() ושיאנו נחוץ יותר.

לדוגמא, אם התמונה המקורית היא בגודל 640\*480, אזי ייווצרו 9 תתי תמונות בגדלים הבאים:

320\*240  
160\*120  
80\*60  
40\*30  
20\*15  
10\*7  
5\*3  
2\*1  
1\*1



שאלה מס' 3 (15 נקודות)

כתבו פונקציה

`char * strrchr(const char * str, char c);`

הפרמטר הראשון הוא מחרוזת (תניח שהמחרוזת מסתיימת ב-\0) והפרמטר השני הוא תו. הפונקציה מחזירה מצביע למופע האחרון של התו במחרוזת. אם התו לא נמצא, הפונקציה תחזיר NULL. אין להשתמש בפונקציות נוספות.

שאלות אמריקאיות

שאלה מס' 4 (10 נקודות)

איזה מהפלטים הבאים אינו פלט אפשרי של התוכנית הנתונה

- ~~0 (a)~~
- 1 (b)
- ~~2 (c)~~
- 3 (d)
- 4 (e)

```
#define N 3
int main(void){
    int i, number, parity=0, x=1, pd[2];

    if (pipe(pd) < 0) exit(1);

    for (i = 0; i < N; ++i){
        if (fork() == 0){
            close(pd[0]);
            if (write(pd[1], &i, sizeof(int)) == -1) exit(2);
            return;
        }

        close(pd[1]);
        for (i = 0; i < N; ++i) {
            if (read(pd[0], &number, sizeof(int)) == -1) exit(3);
            if (parity++ % 2) {
                x *= number;
            } else {
                x += number;
            }
        }
    }

    printf("%d\n", x);
    return 0;
}
```

$\begin{array}{|c|} \hline 1 \\ \hline 2 \\ \hline 1 \\ \hline 0 \\ \hline \end{array} \rightarrow 2$

$\begin{array}{|c|} \hline 1 \\ \hline 2 \\ \hline 0 \\ \hline \end{array}$

✓ 12  
1 0 1

parity	
1	*
2	+
3	*

$$((1 \cdot x_0) + (x_1)) \cdot x_2$$

$$(x_0 + x_1) \cdot x_2$$

2 ✓  
0 ✓

$$(1 \cdot x_0 + x_1) \cdot x_2$$

$$\begin{array}{r} 6 \cdot 1 \cdot 2 \\ (1+2) \cdot 0 \\ (0+1) \cdot 1 \end{array}$$

שאלה מס' 5 (10 נקודות)

נתון ה-makefile הבא:

```
SPECFILE ?=spec.t
CFLAGS=-Wall -pedantic
asm: asm.o inst.o
sim: sim.o
inst.o: inst.h
asm.o: inst.h
sim.o: inst.h
inst.h inst.c: $(SPECFILE) gen.pl
    gen.pl $<
%.e : %.s asm
    asm $<
clean:
    rm -rf inst.o asm.o sim.o asm sim inst.c inst.h *.e
.DELETE_ON_ERROR: *.e
```

כשכותבים 'ls -lt' (פקודה המדפיסה את תוכן ה-directory בסדר יורד של זמן עידכון אחרון), מקבלים:

```
-rw-r--r--  1 efif  math      167 May 23 22:47 spec.t
-rwxr-xr-x  1 efif  math     21k May 23 22:45 sim
-rw-r--r--  1 efif  math     9.6k May 23 22:45 sim.o
-rw-r--r--  1 efif  math      44 May 23 22:45 prog.e
-rwxr-xr-x  1 efif  math     23k May 23 22:45 asm
-rw-r--r--  1 efif  math     11k May 23 22:45 asm.o
-rw-r--r--  1 efif  math     1.8k May 23 22:45 inst.o
-rw-r--r--  1 efif  math     179 May 23 22:45 inst.c
-rw-r--r--  1 efif  math     618 May 23 22:45 inst.h
-rwxr-xr-x  1 efif  math     25k May 23 22:34 gen.pl
-rw-----  1 efif  math     2.3k May 23 22:24 asm.c
-rw-----  1 efif  math     247 May 23 22:24 makefile
-rw-----  1 efif  math     1.2k May 23 22:24 sim.c
-rw-r--r--  1 efif  math      30 May 10 10:54 prog.s
```

מה נקבל כשנכתוב 'make prog.e sim'? בחר תשובה אחת מהבאות.

a)	c)
gen.pl spec.t	gcc -g -c -o asm.o asm.c
gcc -g -c -o asm.o asm.c	gcc -g -c -o inst.o inst.c
gcc -g -c -o inst.o inst.c	gcc -g asm.o inst.o -o asm
gcc -g asm.o inst.o -o asm	asm prog.s
asm prog.s	gcc -g -c -o sim.o sim.c
gcc -g -c -o sim.o sim.c	gcc -g sim.o -o sim
gcc -g sim.o -o sim	
b)	d)
gen.pl spec.t	gen.pl spec.t
gcc -g -c -o asm.o asm.c	gcc -g -c -o asm.o asm.c
gcc -g -c -o inst.o inst.c	gcc -g -c -o inst.o inst.c
gcc -g asm.o inst.o -o asm	gcc -g asm.o inst.o -o asm
asm prog.s	asm prog.s
gcc -g -c -o sim.o sim.c	gen.pl spec.t
gcc -g sim.o -o sim	gcc -g -c -o sim.o sim.c
sim prog.e	gcc -g sim.o -o sim

prog.s

asm.o inst.h

gen.pl

שאלה מס' 6 (10 נקודות)

נתונה התוכנית הבאה:

```
int A[2];
int k;

void foo(int x, int y, int z) {
    z = 0;
    y = 1;
    A[k] = 7;
    x++;
}

int main() {
    A[0] = 2;
    A[1] = 4;
    k = 1;
    foo( A[1], A[k], k);
    printf("A[0]=%d, A[1]=%d, k=%d\n", A[0], A[1], k);
    return 0;
}
```

מה יודפס על המסך ?

- A[0]=7, A[1]=4, k=0 (a)
- A[0]=3, A[1]=7, k=1 (b)
- A[0]=2, A[1]=1, k=1 (c)
- A[0]=2, A[1]=7, k=1 (d)

30

$$\frac{10}{10}$$

הנהגת המוסדות

דבר זה אינו כוונה להפחית את חשיבות ה-Assertion

long root pos;

```

NODE * root;

```

```
root = malloc (sizeof(Node));
```

```
root pos = find root ( input_file );
```

```

fseek(input-file, root_pos, SEEK_SET);

```

```
root = buildTree (rootPos, input-file, root);
```

```
return root; }
```

long find root (FILE \*input\_file)

$$1 + \text{length}$$

inf OK:

```
int cur_pos;
```

```
find(a, length, sizeof(int), 1, inputfile);
```

04 = fseek(input - file, sizeof(int) + sizeof(long) + sizeof(char) + length, SEEK\_CUR);

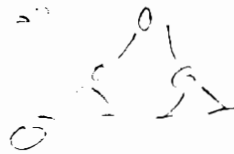
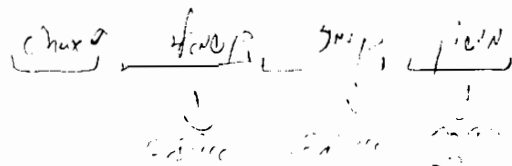
```
if (!ok) return cur_pos;
```

```
find root (input - file);
```

3

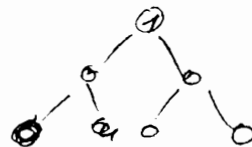
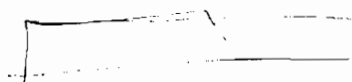


שאלה

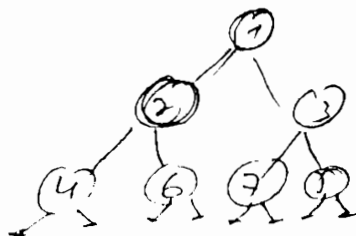


0

0



מסלול



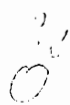
1 6 2 7 5 3 (1)

האלגוריתם של AVL

יציאה

הכנסה

left  
right



האלגוריתם של AVL  
הכנסה - יציאה



המשפט 1.2

NODE \* buildTree( long rootPos, FILE \* ifp, Node \* ifp )

```

int length;
long left_off, right_off;
✓ if (rootPos < 0) return null;
root->left = malloc( sizeof(Node) );
root->right = malloc( sizeof(Node) );

root->left->left = null; root->left->right = null;
root->right->left = null; root->right->right = null;

fread( &length, sizeof(int), 1, ifp );
fread( &left_off, sizeof(long), 1, ifp );
fread( &right_off, sizeof(long), 1, ifp );

if (length) {
    root->Data = malloc( sizeof(char) * length );
    fread( root->Data, sizeof(char), length, ifp );
    buildTree( left_off, ifp, root->left );
    buildTree( right_off, ifp, root->right );
}

return root;
    
```

מחלקת מדעי המחשב

מחלקת מדעי המחשב

מחלקת מדעי המחשב

מחלקת מדעי המחשב



מחלקת מדעי המחשב



מחלקת מדעי המחשב



מחלקת מדעי המחשב



מחלקת מדעי המחשב

typedef unsigned int U\_INT;

$\left(-\frac{1}{2}\right)$

char

each component is an unsigned char!

$\left(\frac{14}{15}\right)$

int insert (U\_INT x, U\_INT y, Image \*img, Image \*subimg) {

int res;

int i, j;

int w0, w1, h0, h1;

if (img && subimg) {

res = img->Data \* img->width \* img->height;

res \*= subimg->Data \* subimg->width \* subimg->height;

if (!res)

return -1; } else return -1;

if (x + subimg->width > img->width

|| y + subimg->height > img->height)

return -1;

w0 = img->width; h0 = img->height; w1 = subimg->width; h1 = subimg->height;

for (i = 0; i != w1; i++)

for (j = 0; j != h1; j++) {

memcpy (&img->Data[(y+j)\*3\*w0+3\*(x+i)], &subimg->Data[(y1+j)\*3\*w1+3\*x1], 3);

}

כל תוכנית  $\left(\frac{1}{2}\right)$

return 0;

}

מסלול 2 מסלול 1

int sold\_insert (Image \*img) {

13/15

int centerX, centerY;

Image temp;

Image \*dst = &temp;

י.ל. להעביר האקס תמונה הקטנה אל תמונה הגדולה

יש להעביר תמונה (ניקו, יאמל) קטנה, ואז לתת לה את התואר

(-1)

res = scale\_image (img, dst);

if (res < 0) return -1;

if (dst->width == 1) res = sold\_insert (dst); // אם רק עמוד אחד יקרא

! מה עם העמוד? (-1) if (res < 0) return -1;

centerX = img->width / 2;

centerX += img->width / 4;

centerY = img->height / 2;

centerY += img->height / 4;

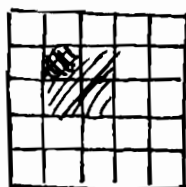
insert (centerX, centerY, img, dst);

free (dst);

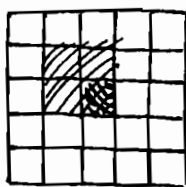
return;

}

הטענה בקוד היא שיש להעביר תמונה קטנה אל תמונה גדולה. 5x5



טענה



טענה

פונקציה

char \* strchr (const char \* str, char c) {

char \* best\_ptr;

char \* ptr;

if (!str[0]) return 0;

// אם הפונקציה

if (str[0] == c)

// אם הפונקציה

best\_ptr = str;

// אם הפונקציה

else best\_ptr = 0;

// אם הפונקציה

ptr = strchr (str + 1, c);

// אם הפונקציה

if (ptr > best\_ptr)

// אם הפונקציה

best\_ptr = ptr;

// אם הפונקציה

return best\_ptr;

// אם הפונקציה

הוא יכול להיות null  
if (ptr == null)  
return 0;  
הוא יכול להיות null  
if (ptr == null)  
return 0;  
הוא יכול להיות null  
if (ptr == null)  
return 0;

מגש ע"י T.A. Marathon

מגש ע"י T.A. Marathon

מגש ע"י T.A. Marathon

מגש ע"י T.A. Marathon