

# מבחן במערכות הפעלה

## ענת ברמלר וסיון טולדו מועד א' סמסטר ב' תשס"ב

### הוראות

יש לענות על כל השאלות.  
יש לצרף את טופס המבחן ודף העזר למחברת הבחינה. מחברת ללא טופס או ללא דף עזר תפסל.  
בשאלות אמריקאיות, יש לסמן את התשובה הנכונה בעיגול על טופס הבחינה ולנמק כשדרש נימוק. בשאלות שבהן יש צורך לנמק, תשובה ללא נימוק לא תזכה באף נקודה.  
יש לענות על כל השאלות בגוף הבחינה במקום המיועד לכך. המקום המיועד מספיק לתשובות מלאות.  
תשובות במחברת הבחינה לא תבדקנה.  
יש למלא מספר ת"ז על כל דף של טופס הבחינה.

אסור השימוש בחומר עזר כלשהו, כולל מחשבוניו או כל מכשיר אחר פרט לעט.

בהצלחה!

### שאלה 1 (24%):

עבור כל אחד מהסעיפים הבאים ענה נכון לא נכון – נמק:

1. בזמן מעבר לטיפול בשגרת פסיקה המעבד עובר למצב מיוחס. נכון/לא נכון

2. ממשקים אחידים לגישה לחומרה מונעים לפעמים שימוש ביכולת מתקדמת של החומרה. נכון/לא נכון

3. DMA מונע למעשה את הצורך בהעברת פקודות לבקר באמצעות פקודות מכונה. נכון/לא נכון

4. בלינוקס לא ניתן ליצור מצביע סימבולי אל מדריך על מנת שלא ייווצרו מעגלים במרחב השמות. נכון/לא נכון

5. עליך לרכוש כרטיסי רשת (network interface cards) או כרטיסי הרחבה עם בקר לרשת תקשורת) עבור שני מחשבים. בחנות המחשבים המוכר הציע לך שני כרטיסים, אחד מתוצרת אוסם והשני מתוצרת תלמה. המוכר אמר לך שזכור לו שאחד המחשבים מיועד לשרתים והשני מיועד לתחנות עבודה, אבל הוא לא זכר איזה כרטיס מתאים לאיזה סוג תצורות. הוא כן ידע לומר שלקוחות רבים קנו כרטיסים משני הסוגים וכולם מרוצים מאוד. בבדיקה מעמיקה של הכיתוב על האריזה התברר לך שההבדל היחיד בין הכרטיסים הוא שהכרטיס מתוצרת אוסם מבצע פסיקה לאחר קבלת כל חבילה, ואילו הכרטיס מתוצרת תלמה מבצע פסיקה רק לאחר שמצטברות בזכרון הפנימי של הכרטיס 100 חבילות, או לאחר שעברה שניה מאז שהגיעה החבילה הישנה ביותר בזכרון הפנימי של הכרטיס.

לנתב שצריך להעביר כמויות גדולות של תעבורה תבחר את כרטיס תלמה ולתחנת עבודה שצריכה לספק למשתמש זמן תגובה קצר תבחר את כרטיס אסם. נכון / לא נכון

---



---



---

6. Raid 3 יותר אמין מ-Raid 1 (כלומר הסיכוי שלאחר נפילה או שורה של נפילות לא נוכל לשחזר מידע, קטן יותר ב-Raid 3).

---



---

### שאלה 2 (35%):

נתונות ארבעת תוכניות הבאות הכתובות ב-pseudo code דמוי C. התוכניות משתמשות בפונקציות הבאות:  
 createWindow() - המייצרת חלון חדש שבו רץ התהליך או החוט שקרא לתוכנית.  
 GetHrTimer() – המחזירה את מספר ה-cycles שמחשב ביצע מתחילת פעולתו.

עבור כל אחת מהתוכניות רשום בקצרה את כל האפשרויות השונות של פלט הנוצר בכל אחד מחלונות התהליכים או החוטים הנוצרים. נמק בקצרה.

.א

```
int time;

main() {
    int k=0;
    id=fork()

    createWindow();
    time=GetHrTimer();
    while (k<100) {
        k++;
        printf(" %d ",time);
    }
}
```

---



---



---

.ב.

```
int time;

main()
{
    CreateThread(, , .printTime(), .);
    CreateThread(, , .printTime(), .);
}

printTime() {
    int k=0;
    createWindow();
    time=GetHrTimer();
    while (k<100) {
        k++;
        printf(" %d ",time);
    }
}
```

---

---

---

.ג.

```
main() {
    createThread(, , printTime(), .);
    createThread(, , printTime(), .);
}

printTime() {
    int time;
    int k=0;

    createWindow();
    time=GetHrTimer();
    while (k<100) {
        k++;
        printf(" %d ",time);
    }
}
```

---

---

---

.ד

```

int k=0;

main()
{
    createThread(,,printTime(),..);
    createThread(,,printTime(),..);
}

printTime()
{
    int time;

    createWindow();
    time=GetHrTimer();
    while (k<100) {
        k++;
        printf(" %d ",time);
    }
}

```

---



---



---

**שאלה 3 (29%):**

תוכנית ממיינת מספרים שמורים במערך של משתנים מסוג double שכל אחד מהם תופס 8 בתים. המיין מתבצע באלגוריתם mergesort. האלגוריתם ממיין באופן רקורסיבי את החצי הראשון של המערך, לאחר מכן את חלקו השני, ולבסוף ממזג את החצאים הממויינים. נניח שמחשב עם 128Mbytes של זכרון אמיתי ו backing store בגודל 2048Mbytes ממיין מערך בגודל 256Mbytes, ושגודל דפים ומסגרות במערכת ההפעלה הוא 8192 בתים. נניח גם שהעברת מסגרות בין הזכרון האמיתי ובין ה backing store מתבצע במדיניות LRU. א. כמה דפים (לא בהכרח שונים) יועברו מה backing store לזכרון האמיתי במהלך המיין אם המערך ממויין כבר? התעלם/התעלמי מהמסגרות של זכרון אמיתי שהקוד של התוכנית צורך ומהמסגרות שמערכת ההפעלה או תוכניות אחרות צורכות (כלומר יש להניח שכל המסגרות של הזכרון האמיתי בשימוש עבור המערכים של אלגוריתם המיין). תזכורת: 1Mbyte הוא  $1024*1024$  בתים.

---



---



---



---



---



---

ב. כיצד תשתנה תשובתך לסעיף א, כאשר אלגוריתם המיון יהיה bubblesort.

---



---



---



---



---

לנחיותך, מצורף כנספח לשאלה זו הקוד של אלגוריתמי המיון, באם אינך זוכר את האלגוריתמים הבסיסים של מיון מערכים,

**שאלה 4 (12%):**

א. אזרחית בשם מושית בת-שבע התחברה לאינטרנט המהיר בעזרת ADSL מספק האינטרנט שלה הובהר לה שהחיבור מספק לה כתובת IP אחת בלבד, ושאם תרצה ביותר כתובות IP על מנת לחבר לאינטרנט את כל שבעת המחשבים שבביתה עליה לשדרג את החיבור שלה לחבילת ULTRA-SUPER-EXTRA שעלותו גבוהה מעט יותר. האם מושית יכולה לחבר אם כל המחשבים לאינטרנט בלי לשלם יותר, אם כן איך ואם לא למה?

---



---

ב. איזה שירות יקיצה (timer) מונע מ-TCP למנוע משולח לעד לשלוח מידע בגלל אובדן עדכון חלון?

---



---

בהצלחה !!!

נספח א – Merge sort קוד:

```

void merge_sort(int data[], int first, int last) {
    int* temp = malloc((last - first + 1) * sizeof(int));
    recursive_msort(data, temp, first, last, 0);
}

void recursive_msort(int data[], int temp[],
                    int first, int last,
                    int depth) {
    if (first == last) {
        if (depth % 2 == 1) temp[first] = data[first];
    } else {
        int half = (last - first) / 2;
        recursive_msort( data, temp, first, half, depth+1 );
        recursive_msort( data, temp, half+1, last, depth+1 );
        merge( data, temp, first, half, last, depth );
    }
}

void merge(int data[], int temp[],
          int first, int half, int last,
          int depth) {
    if (depth % 2 == 1) { // input in temp, out in data
        int i=first;
        int j=last+1;
        int k;
        for (k=first; k<last; k++) {
            if (j>last || ( i<=half) && temp[i] < temp[j] ) {
                data[k] = temp[i];
                i++;
            } else {
                data[k] = temp[j];
                j++;
            }
        }
    } else { // input in data, output in temp
        . . . // similar
    }
}

```

נספח ב – Bubble sort קוד:

```

void bubble_sort(int a[], int n) {
    int i,j;

    for ( i=0;i< n-1; i++)
        for (j=n-1;j>i; j--)
            if ( a[j-1] > a[j] )
                swap( &a[j-1], &a[j]);
}

void swap( int *x, int *y) {
    int tmp;

    tmp = *x;
    *x = *y;
    *y = tmp;
}

```