בחינה בעקרונות מערכות הפעלה

קראו בעיון לפני שתתחילו בפתרון הבחינה!

1. המבחן מורכב משלושה חלקים.
2. בחלקים א' ו ב' מופיעות שאלות פתוחות. ענו **תשובות מלאות**, **בכתב קריא** ו**בקיצור נמרץ**. אין חובה להשתמש בכל השורות המוקצות לצורך התשובות, אך אין לחרוג מהמקום המוקצה.
3. בחלק ג' (שאלות אמריקאיות) עליכם לבחור בכל פעם בתשובה יחידה מבין התשובות המוצעות ולהקיף בעיגול את אות התשובה שבחרתם.
4. כל חומר עזר אסור לשימוש בזמן הבחינה פרט למחשבון פשוט
5. משך הבחינה - שלוש שעות.

**את התשובות לכל השאלות יש לכתוב בשאלון הבחינה.**

בהצלחה!

## חלק א (55 נקודות)

ענו על **שלוש** השאלות הבאות.

### שאלה 1 (15 נקודות)

|  |  |  |
| --- | --- | --- |
| (10 נק') | א. | עליכם לממש רשימה מקושרת חד כיוונית ציקלית (מעגלית). הרשימה תהיה בעלת איבר סרק (dummy node) יחיד, שמוצבע מראש הרשימה. בצורה זאת אין צורך לבדוק את מקרה הקצה של הוצאת איבר אחרון מהרשימה או הוספת איבר ראשון – ברשימה תמיד יש איבר אחד.  השלימו את הקוד הבא להכנסת איבר לראש הרשימה. עליכם להשלים גם את מבני הנתונים המגדירים את הרשימה ואיבר הרשימה. ניתן להשתמש במנעול אחד בלבד. קוד לא יעיל (מבחינת מספר המשתנים או הביצועים) יקבל ניקוד חלקי בלבד.  typedef struct Node\_t {  void\* data;  Node \*next;  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  } Node;  typedef struct List\_t {  Node head; // dummy empty node. Always first  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  } List;  void insert(List list, void\* data){  Node\* node = (Node\*)malloc(sizeof(Node));  node->data=data;  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  } |
| (15 נק') | ב. | נסתכל על מימוש של פונקציית remove מרשימה מקושרת דו כיוונית ציקלית (מעגלית). הרשימה נועדה לשפר את הגישה המקבילית של ממספר תהליכונים לרשימה ולכן משתמשת במנעולים ברמה של איברי הרשימה.  typedef struct Node\_t {  void\* data;  Node \*next, \*prev;  Lock lock;  } Node;  typedef struct List\_t {  Node head, tail; // dummy empty nodes  } List;  InitList(){  head->prev=head->next=tail;  tail->prev=tail->next= head;  }  void remove(List\* list, Node\* node)  {  Node\* prev;  if (node==NULL || node->prev==NULL || node->next==NULL) return;    lock(node->prev->lock);  lock(node->lock);  lock(node->next->lock);  node->prev->next = node ->next;  node->next ->prev = node ->prev;  prev=node->prev;  node->next=NULL;  node->prev=NULL;  unlock(prev ->next->lock);  unlock(node->lock);  unlock(prev->lock);  }  במימוש זה קיימת בעיה. מצאו את הבעיה והסבירו מדוע היא יכולה להיגרם ובאילו תנאים.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| (10 נק') | ג. | כעת נחליף את פונקצית הremove מסעיף ב בפונקציה הבא:  void remove(List\* list, Node\* node)  {  Node\* prev;  if (node==NULL) return;  lock(node->lock);  if (node==NULL || node->prev==NULL || node->next==NULL){  unlock(node->lock);  return;  }  lock(node->prev->lock);  lock(node->next->lock);  node->prev->next = node ->next;  node->next ->prev = node ->prev;  prev=node->prev;  node->next=NULL;  node->prev=NULL;  unlock(prev ->next->lock);  unlock(node->lock);  unlock(prev->lock);  }  הוכיחו או הפריכו את ייתכנות ה deadlock בעקבות השימוש בפונקציה זו.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

### שאלה 2 (15 נקודות)

במערכת הפעלה כלשהי קיים מנגנון לניהול זיכרון המקצה לכל **דף** מזהה מספרי i. מדיניות הדפדוף הממומשת ע"י המנגנון לניהול הזיכרון היא המדיניות הבאה:

אם נרצה להביא לזיכרון דף מספר i, והזיכרון מלא, נפנה מהזיכרון דף j כך ש |j-i| הינו מקסימאלי. אם יש שני דפים כאלה נבחר אחד באקראי.

בשאלה זו נניח שגודל הזיכרון הפיזי הוא 3 מסגרות (אלא אם מצוין אחרת).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| (5 נק') | א. | עבור סדרת הגישות הבאה, רשמו את תוכן הזיכרון. כמה Page Faults מתבצעים במהלך הריצה? (המספרים מציינים את מזהי הדפים). הניחו שהזיכרון היה ריק בתחילת הריצה.  1 2 3 4 1 4 3 2 1 4 (משמאל לימין)   |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **t** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | | frame 1 |  |  |  |  |  |  |  |  |  |  | | frame 2 |  |  |  |  |  |  |  |  |  |  | | frame 3 |  |  |  |  |  |  |  |  |  |  |   סה"כ: page faults **\_\_\_\_** |
| (5 נק') | ב. | תנו דוגמא לסדרת גישות בה אלגוריתם LRU טוב יותר ממדיניותה של המערכת.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| (5 נק') | ג. | הוכיחו או הפריכו את הטענה הבאה: אלגוריתם LRU תמיד טוב יותר ממדיניותה של המערכת.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| (5 נק') | ד. | השוואו בין כמות המסגרות אותה יש לבדוק כדי למצוא דף לפינוי מהזיכרון, לפי האלגוריתם LRU ולפי מדיניותה של המערכת.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

## חלק ב (25 נקודות)

ענו על **חמש** השאלות הבאות. משקל כל שאלה 5 נקודות.

### שאלה 3

מהו hard-link? מה ההבדל בינו לבין soft-link?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### שאלה 4

מהו condition variable וכיצד משתמשים בו במבנה פיקוח (monitor) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### שאלה 5

מהי TRAP instruction?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### שאלה 6

מהו inverted page table? ציירו כיצד מתבצע תרגום כתובת לוגית לכתובת פיזית באמצעות inverted page table.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### שאלה 7

כיצד נראה מבנה של קובץ ספרייה במערכת קבצים ext2? שרטטו והסבירו. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

שרטוט:

## חלק ג (20 נקודות)

ענו על **ארבע** שאלות רב-ברירה (אמריקאיות). משקל כל שאלה 5 נקודות.

בכל שאלה יש לבחור את התשובה הנכונה ולהקיף בעיגול את אות התשובה שבחרתם.

### שאלה 8

במערכת עם ניהול זיכרון באמצעות הדפדוף (paging) נתון כי:

* כתובת מדומה (virtual address) הינה בת 64 bits
* גודל הדף הוא 64 Kbytes
* גודל המילה הוא 4 bytes

מהי כמות הדפים המכסימלית בזיכרון המדומה?

א) 2^40

ב) 2^50

ג) 2^54

ד) 2^60

### שאלה **9**

מהו מספר הכניסות (entries) בטבלת הדפים המהופכת (inverted page table)?

א) כמספר הדפים בזיכרון המדומה

ב) כמספר המסגרות (frames) בזיכרון הפיזי

ג) כמספר הדפים המכסימלי שיכול לדרוש תהליך

ד) כמספר התהליכים המכסימלי שיכולים להתבצע בו-זמנית במערכת

### שאלה **10**

כדי לבחור את הדף המתאים לפינוי מהזיכרון הראשי הוצע האלגוריתם הבא:

לכל מסגרת (frame) בזיכרון מוצמד מונה. כאשר דף חדש מגיע למסגרת, ערכו של מונה זה נקבע להיות 0 ובכל פניה לדף המונה מועלה ביחידה אחת. כאשר יש צורך לפנות דף מהזיכרון, בוחרים תמיד את הדף שהמונה המוצמד למסגרת שלו מכיל את הערך הנמוך ביותר. האם אלגוריתם זה הוא טוב לניהול הדפדוף בזיכרון?

א) כן. המונה משקף את מידת נחיצותו של הדף. ככל שערכו נמוך, כך מספר הפניות לדף היה נמוך יותר ולכן הוא מתאים לפינוי.

ב) כן. מונה גבוהה מצביע על דף הנמצא בשימוש מתמיד.

ג) לא. אלגוריתם זה אינו מאפשר כלל כניסה של דפים חדשים כי המונה שלהם תמיד 0 ולכן יפונו מיד.

ד) לא. מצב המונה עלול לשקף מציאות היסטורית שאיננה קיימת עוד.

### שאלה **11**

אחד ההבדלים בין פונקציות ספריה בשפת C (כגון printf()) לבין קריאות מערכת (system calls) ב- Unix/Linux הוא:

א) רק למנהל המערכת (super user) מותר להשתמש בקריאות מערכת, בפונקציות ספריה יכול להשתמש כל אחד

ב) באמצעות פונקציות הספרייה בשפת C לא ניתן לפתוח קבצים לפעולות קריאה או כתיבה

ג) כל קריאות המערכת הן חלק מפונקציות הספרייה של השפה

ד) כל פונקציות הספרייה משתמשות בקריאות מערכת

ה) התשובות ג' וד' הן הנכונות

ו) כל התשובות הקודמות אינן נכונות

**בהצלחה!**