**מבחן במערכות הפעלה אולטרא קוד 2024 ב'**

1. (5 נקודות) איזו פעולה מבין הפעולות הבאות אפשר לבצע אך ורק במצב ה kernel mode ?

1) שינוי הרשאות קבצי המערכת.

2) החלפה בין user threads .

3) גישה למרחב כתובות של תהליך אחר.

4) השמת ערך לתוך משתנה קיים באזור זיכרון משותף .

5) יצירת משתמש חדש במערכת.

2. (5 נקודות) אלגוריתם הבנקאי הוא:

* 1. אלגוריתם המונע deadlock על ידי שבירת האפשרות למעגל בגרף התלויות.
  2. אלגוריתם המונע deadlock על ידי הכתבת סדר חד משמעי לביצוע בקשות של תהליכים שונים.
  3. אלגוריתם המתחמק מ deadlock על ידי בדיקה, שבכל שלב קיים סדר כלשהו המאפשר הרצה וסיום כל התהליכים הקיימים במערכת.
  4. אלגוריתם המאפשר להיחלץ מצב deadlock עם פגיעה מינימלית בתהליכים.
  5. אלגוריתם שמונע deadlock על ידי הגבלה מראש על דרישות משאבים של כלל תהליכים.

3. (5 נקודות) מה התיאור המתאים ביותר לטיפול בפסיקת חומרה?

1) פסיקה מתקבלת ישירות ע"י גרעין מערכת ההפעלה ומטופלת על ידו.

2) פסיקה מתקבלת ע"י תהליך המשתמש ומועברת לגרעין מערכת ההפעלה.

3) פסיקה מתקבלת ע"י המעבד, שמתחיל לבצע קטע קוד טיפול בפסיקה, בתוך הגרעין.

4) פסיקה מתקבלת ע"י המעבד לבצע קטע קוד טיפול בפסיקה בתוך במרחב משתמש.

5) התיאור תלוי בסוג פסיקת חומרה.

בשלוש השאלות הבאות, בשביל כל אחת מבעיות ה synchronization , בחרו את אמצעי ה synchronization הטוב (היעיל) ביותר בכדי לפתור את הבעיה. אין צורך לממש אלגוריתם שלם. **באלגוריתם יכולים להיות מבני נתונים/משתנים שהכרחיים לפתרון, אין צורך להתייחס אליהם.**

אין צורך לדאוג לאפשור גישה לאמצעי ה synchronization, צריך להניח שיש גישה.

4. (5 נקודות) צוות מתכנתים מפתח מערכת הזמנת כרטיסי טיסה המאפשרת ביצוע הזמנה במקביל. כרטיסים מוצעים בחבילות של 30, מתי שלא נשאר שום כרטיס פנוי, מביאים מטוס גדול יותר. הזמנה מתייחסת למקום בטיסה ולא למקום מושב ספציפי.

כל המתחבר למערכת מטופל כ THREAD נפרד ואם אין מקום, ה THREAD עובר למצב המתנה. כשמספר מקומות גדל, ה THREAD מתעורר. מה אמצעי synchronization שיאפשר פתרון יעיל ביותר?

1) Counting semaphore

2) Binary semaphore

3) Lock and Condition variable with Condition signal

4) Lock and Condition variable with Condition broadcast

5) Lock

5. (5 נקודות) במשחק מחשב קיימת בעיית משקל יתר של מכוניות בגשר בכביש חד-כיווני. המכוניות נוסעות במהלך משחק מחשב, כך שכל מכונית היא THREAD נפרד. בסה"כ במשחק יש 6 מכוניות רגילות ו 2 מכוניות משא. צריך לבחור אמצעי synchronization שיבטיח/יבטיחו שבכל נקודת זמן לא יהיו יותר מ 3 מכוניות רגילות בסה"כ או מכונית משא אחת(לבד) על הגשר. האמצעי/אמצעים חייב/חייבים להיות היעיל ביותר.

1) One Mutex

2) Two Mutexes

3) Three Mutexes

(4Two Mutexes and two Binary Semaphores

5) One Counting semaphores

6. (5 נקודות) במשחק מחשב קיימת בעיית תנועה בקטע כביש צר חד-נתיבי שהתנועה בו מותרת בכל רגע נתון רק בכיוון אחד. כל עוד שיש תנועה באותו הכיוון, מכוניות נוספות יכולות לנסוע באותו הכיוון. אבל מכוניות שרוצות לנסוע מהצד השני של הקטע הצר בכיוון הפוך, צריכות להמתין עד שהתנועה הקיימת תיפסק. המכוניות נוסעות במהלך משחק מחשב כשכל מכונית היא THREAD נפרד. צריך לבחור אמצעי synchronization שיבטיח/יבטיחו את פתרון הבעיה. האמצעי/אמצעים חייב/חייבים להיות היעיל ביותר.

1) One Counting semaphore

2) Two Mutexes

3) Three Mutexes

(4Two Mutexes and two Binary Semaphores

5) Two Counting semaphores

7 .(10 נקודות) נתונה התוכנית הבאה (pseudo code) שמיועדת ל SYNCHRONIZATION בין מספר כלשהו של THREADS שרצים במקביל. כל THREAD מבצע אותו פרוטוקול כניסה ויציאה ל CRITICAL SECTION.

התוכנית משתמשת ב N Mutexesשאותחלו ל 1 ו Mi הוא שם של Mutex,

בגלל שמדובר בפסודו-קוד זה כמו M1 , 2M , ... כאשר N הוא מספר טבעי גדול מ- 2.

שני הפרוטוקולים דומים, רק Mutexes משתחררים אחרי סיום ה CRITICAL SECTION בסדר שונה.

|  |  |
| --- | --- |
| **Protocol 2** | **Protocol 1** |
| while (1){  for (i=0; i++; i<N)  Lock (Mi);  /\* Critical section \*/  for (i=N-1; i--; i>=0)  Unlock(Mi);  } | while (1){  for (i=0; i++; i<N)  Lock (Mi);  /\* Critical section \*/  for (i=0; i++; i<N)  Unlock(Mi);  } |

בחרו בטענה הנכונה לגבי שני הפרוטוקולים 1 ו- 2 :

1) שני פרוטוקולים פותרים בעיית קטע קריטי ויעילים באותה מידה.

2) פרוטוקול 2 פותר בעיית קטע ופרוטוקול 1 יכול לגרום ל קיפאון DEADLOCK .

3) פרוטוקול 1 פותר בעיית קטע ופרוטוקול 2 יכול לגרום ל קיפאון DEADLOCK .

4) פרוטוקול 2 פותר בעיית קטע ופרוטוקול 1 לא מבטיח מניעה הדדית בקטע קריטי.

5) שני הפרוטוקולים פותרים בעיית קטע קריטי ופרוטוקול 2 יעיל יותר מפרוטוקול 1.

8.(10 נקודות) מערכת הקבצים של מערכת הפעלה מסוימת משתמשת בשיטת ה I-node עם פרמטרים הבאים:

* גודל הבלוק במערכת הקבצים הוא Kbytes 4
* כתובת הבלוק היא 4 בתים (bytes)
* 10 שדות של ה I-node יכולים להחזיק ישירות כתובת הבלוק בדיסק
* שדה נוסף אחד נועד להחזיק כתובת של ה single indirect block
* עוד שדה נוסף אחד נועד להחזיק כתובת של ה double indirect block
* ועוד שדה נוסף אחד נועד להחזיק כתובת של ה triple indirect block

חשבו מה הגודל המרבי של קובץ שניתן לאכסן ב 3076 בלוקים בסה"כ(כולל נתונים ומצביעים, אבל לא כולל את ה I-NODE עצמו) .

הערה: לצורך החישוב 1 MByte = 1024 KByte , 1 KByte = 1024 Byte

1. Kbytes 12292
2. Kbytes 12288
3. Kbytes12328
4. Kbytes12332
5. Kbytes12336

9.(5 נקודות) מה הבדל בין קישור רך (קיצור דרך, SOFT LINK) וקישור קשיח (HARD LINK)?

1) ל SOFT LINK ולקובץ שהוא מקשר יש I-NODE ים שונים ול- HARD LINK ולקובץ אותו I-NODE

2) פתיחת הקובץ דרך SOFT LINK מהירה יותר בגישה דרך HARD LINK

3) יכול להיות ש HARD LINK מצביע על הקובץ שכבר נמחק מה שלא יכול להיות עם SOFT LINK

4) יצירת HARD LINK דורשת יצירת קובץ נוסף שיכיל אותו וב- SOFT LINK לא דורשת

5) יצירת HARD LINK אפשרית רק בתיקיה שונה מתיקיה שבה נמצא הקובץ ויצירת SOFT LINK

אפשרית גם בתוך אותה תיקיה

10. (10 נקודות) להלן טבלת הדפים של תהליך במערכת עם זיכרון וירטואלי בנקודת זמן מסוימת.

כל המספרים הם דצימליים, מתחילים מאפס, וכל הכתובות הן כתובות של ביית בזיכרון.

גודל הדף הוא 2048 בתים( KByte2 ).

|  |  |  |
| --- | --- | --- |
| Frame Number | Valid bit | Page Number |
| 1 | 1 | 0 |
| 2 | 1 | 1 |
| 1 | 0 | 2 |
| 3 | 1 | 3 |
| 3 | 0 | 4 |
| 0 | 1 | 5 |

לאיזו כתובת פיזית, אם יש כזו, ימופה הכתובת הוירטואלית הבאה: 5000 ?

1) 2952

2) 5000

3) 9096

4) 2953

5) לא ניתן לחשב

11. (5 נקודות) האם יש שגיאות בטבלת הדפים משאלה 10 ואם כן, מה הן? נא לסמן את התשובה המדויקת ביותר.

1) אין שגיאות

2) יש- דף 5 נמצא במסגרת 0 שמספרה נמוך יותר ממספר המסגרת שדף 0 נמצא בה

3) יש רק שגיאה אחת- דף 0 ודף 2 נמצאים(נוכחים) במסגרת עם אותו מספר, הדבר אינו תקין

4) יש שגיאה אחת - דף 3 נמצא(נוכח) במסגרת עם מספר ששווה למספר הדף, הדבר אינו תקין בזיכרון וירטואלי

5) יש 2 שגיאות- גם דף 0 וגם דף 2 נמצאים(נוכחים) במסגרת עם אותו מספר, הדבר אינו תקין

12. (5 נקודות) ניתן ליצור איזור של זיכרון משותף SHARED MEMORY (אזור זיכרון הנוצר ע"י מערכת הפעלה עם הרשאות גישה לכמה תהליכים) בין שני תהליכים PROCESSES ע"י כך שבכל טבלת הדפים יהיו מצביעים לאזור המשותף של שני התהליכים. במצב זה:

1. האזור המשותף חייב להיות ממופה(משויך) **לאותה כתובת וירטואלית** במרחבי הכתובות של שני התהליכים.
2. כאשר אחד התהליכים פונה לכתובת משותפת ויש page fault , אזי שני התהליכים יחסמו **בוודאות**.
3. קידום ערך המשתנה הנמצא באזור המשותף צריך להיות מוגן ע"י סמפור (SEMAPHORE).
4. כל גישה לאזור המשותף תבוצע ע"י מערכת ההפעלה ולא ישירות ע"י התהליך.
5. גודל טבלת הדפים של 2 תהליכים חייב להיות זהה.

13. (5 נקודות) כמה פעמים בסה"כ יודפס "My son" בעקבות הרצת קובץ עם הקוד הבא (אחרי קימפול)?

כל קריאות מערכת מצליחות (לא נכשלות). תזכורת: קריאת מערכת FORK() מחזירה 0 לתהליך בן וערך ששווה ל PID של הבן לתהליך אב.

main() {

fork();

if (fork()==0) { printf("New proc"); }

if (fork()!=0) { printf("My son"); }

}

1) 4

2) 3

3) 8

4) 2

5) 1

14. (5 נקודות) איזו אחת מהבעיות הבאות קיימת כאשר משתמשים בתהליכוני משתמש

user threads ?

* 1. חסימת תהליכון אחד תגרום לחסימת כל התהליך.
  2. החלפה בין תהליכונים (context switch) דורשת מעבר לגרעין המערכת.
  3. לא קיימת אפשרות יצירת משתנים לוקאליים לתהליכון.
  4. לא קיימת אפשרות לשלוט על כמות הזמן שכל תהליכון מקבל.
  5. לא קיימת אפשרות יצירת משתנים גלובאליים משותפים לכל תהליכונים.

15. (5 נקודות) האם יכול להיות מעבר ישיר של תהליך ממצב "מוכן" למצב "חסום"?

1) כן, כאשר התקן שהתהליך צריך יתפנה

2) כן, כאשר התהליך אינו מסוגל לפעול עד אשר משימה כלשהי תושלם

3) לא ניתן לקבוע חד משמעית כי זה תלוי במספר ליבות במעבד

4) לא, תהליך קודם צריך לעבור מצב "רץ"

5) לא, בגלל מדיניות של כל מערכות הפעלה

16.(10 נקודות) קטע פסודו-קוד הבא מיצג ניסיון פתרון בעיית יצרן-צרכנים (יצרן אחד, כמה צרכנים).

**Producer (put)**

lock(&L)

if (state == NOT\_EMPTY)

cond\_wait(&cv, &L)

produce and put in buffer

state = NOT\_FULL

unlock(&L)

cond\_signal(&cv)

**Consumer (get)**

lock(&L)

if (state != NOT\_EMPTY)

cond\_wait(&cv, &L)

consume

state = EMPTY

unlock(&L)

cond\_signal(&cv)

מהו המשפט הנכון לגבי הפתרון המוצע?

1) הפתרון תקין

2) הפתרון יכול לגרום להעדר התקדמות LIVELOCK של יצרן

3) שני צרכנים יכולים לקחת אותו פריט

4) יכול להיות מצב שפריט במאגר יוחלף בפריט אחר בלי שצרכן לקח אותו

5) הפתרון אינו תקין, חוץ מזה כל שאר התשובות נכונות