מה קובע את סוגי ה- system calls ואת מספרם במערכת מחשב?

|  |  |  |  |
| --- | --- | --- | --- |
| סוג המעבד | 0.00% | 1 | 2.86% |
| סוג מערכת ההפעלה (למשל LINUX או WINDOWS) | 100.00% | 33 | 94.29% |
| מנהל המערכת ADMINISTRATOR | 0.00% | 1 | 2.86% |
| גודל הזיכרון הראשי הקיים במערכת | 0.00% | 0 | 0.00% |
| סוג הקומפיילר שבו משתמשים | 0.00% | 0 | 0.00% |
| [אין תשובה] | 0.00% | 0 | 0.00% |

איזה מבין התנאים הבאים מבטיח את מניעת deadlock במערכת עם שלושה PROCESSES לכל היותר ועם שישה סוגים שונים של  RESOURCES

|  |
| --- |
| כל PROCESS יוכל לבקש לא יותר משני  RESOURCES |
| כל PROCESS ישחרר את כל ה- RESOURCES שהוא החזיק בהם (כולל אמצעי סנכרון) לפני שיסתיים . |
| כל ה- PROCESSES אינם מכילים לולאה |
| אחרי ש- PROCESSES קיבל מספר כלשהו של RESOURCES , הוא ישחרר את כולם בתוך 20 שניות לכל היותר ואם צריך, יבקש אותם מחדש. |
| כל תהליך שישהה במצב המתנה יותר מ 20 שניות ברצף, ייהרג ע"י המערכת ולא ייווצר מחדש. |

צוות מתכנתים מפתח מערכת מכירת כרטיסים דיגיטליים למופע מבוקש מאד עם מספר מקומות מוגבל. המערכת אמורה לאפשר לכל המעוניינים ביצוע מקבילי של הפעולות הבאות:

1) רכישת כרטיס ותשלום עליו.

2) ביטול רכישת כרטיס וקבלת הכסף עבורו בחזרה.

בגלל שמדובר במופע מבוקש, כל מתחבר למערכת מטופל כ- THREAD נפרד, וכאשר כבר אין מקום פנוי, ה- THREAD עובר למצב המתנה והלקוח לא מתנתק מן המערכת אלא מחכה עד ש(אולי) יתפנה מקום.

בשלב הראשון הוחלט לממש את המערכת מבלי לדאוג להוגנות FAIRNESS (המשמעות של FAIRNESS היא: מי שמנסה לקנות כרטיס ומועבר למצב ההמתנה משום שלא הצליח, מיקומו בתור יישמר לכשיתפנה מקום למופע ותתאפשר הרכישה).

הציעי אמצעי synchronization שיאפשר פתרון יעיל לדרישות השלב הראשון, מבין האפשרויות הבאות:

מודל תגובה

קרדיט חלקי

ספירה

תדירות

One Counting semaphore0.00%00.00%Binary semaphore0.00%00.00%Lock (without a Condition variable)0.00%00.00%Lock and Condition variable with Condition signal100.00%35100.00%Lock and Condition variable with Condition broadcast0.00%00.00%[אין תשובה]0.00%00.00%

\

כחלק מפיתוח תוכנת הדמיית תנועה בכביש יש צורך לדמות את זרימת התנועה בכביש חד- נתיבי דו-כיווני (נתיב אחד בכל כיוון). הכביש עובר באזור הררי עם אפשרות של מפולות סלעים. כל מכונית ממומשת כ- THREAD נפרד.

בשלב הראשון של הפיתוח יש צורך לאפשר שתי פעולות:

1) עצירת כל המכוניות בקטע מסוים של הכביש ב- 2 הכיוונים כשיש סכנת מפולת סלעים.

2) התחלה בו זמנית של תנועת כל המכוניות כשהסכנה נגמרת.

הציעי אמצעי synchronization שיאפשר פתרון יעיל לדרישות השלב הראשון, מבין האפשרויות הבאות:

| **מודל תגובה** |
| --- |
| Counting semaphore> |
| Binary semaphore |
| Lock (without a Condition variable) |
| Lock and Condition variable with Condition signal> |
| Lock and Condition variable with Condition broadcast> |
| [אין תשובה] |

**שאלה 5 (5 נקודות)**

 במשחק מחשב קיימת בעיית מעבר בגשר צר עם נתיב אחד בלבד (מספר מכוניות יכולות לנסוע בו באותו כיוון, אבל לא בכיוונים שונים). המכוניות נוסעות במהלך המשחק כך שכל מכונית היא THREAD נפרד.

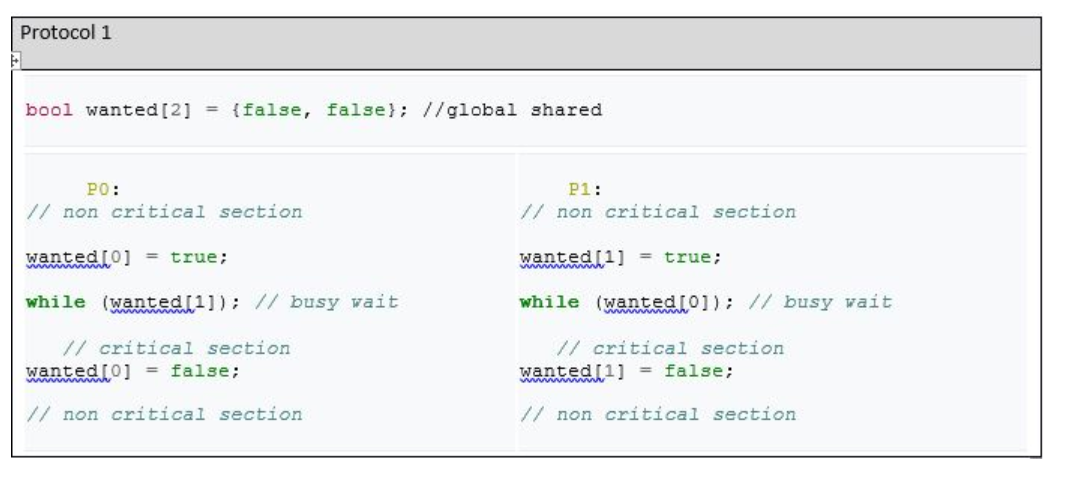
כשמכוניות שנוסעות בכיוון מסוים, מכוניות נוספות יכולות לנסוע באותו כיוון באותו הזמן, אך מכוניות שרוצות לנסוע בכיוון הנגדי צריכות להמתין עד שהגשר יהיה ריק. אין צורך לדאוג למניעת התנגשויות בין המכוניות שנוסעות באותו כיוון.

הציעי כיצד ניתן לפתור בקלות את הבעיה בעזרת אמצעי synchronization מבין האפשרויות הבאות. האמצעי/אמצעים חייב/חייבים להיות היעיל ביותר. עקרונית, בפתרון המוצע, מותר להשתמש בנוסף גם במשתנים רגילים אך צורך להתייחס לזה

| **מודל תגובה** |
| --- |
| Counting semaphore> |
| Binary semaphore |
| 2 Counting semaphore |
| 2 MUTEX (LOCK) |
| Lock and Condition variable with Condition broadcast |
| [אין תשובה] |

**שאלה 6  (5 נקודות)**

נתונה התכנית הבאה (pseudo code) המיועדת ל-  SYNCHRONIZATION בין שני PROCESSES - P0 and P1 הרצים במקביל. כל PROCESS מבצע את אותו פרוטוקול כניסה ויציאה ל/מ-  CRITICAL SECTION.

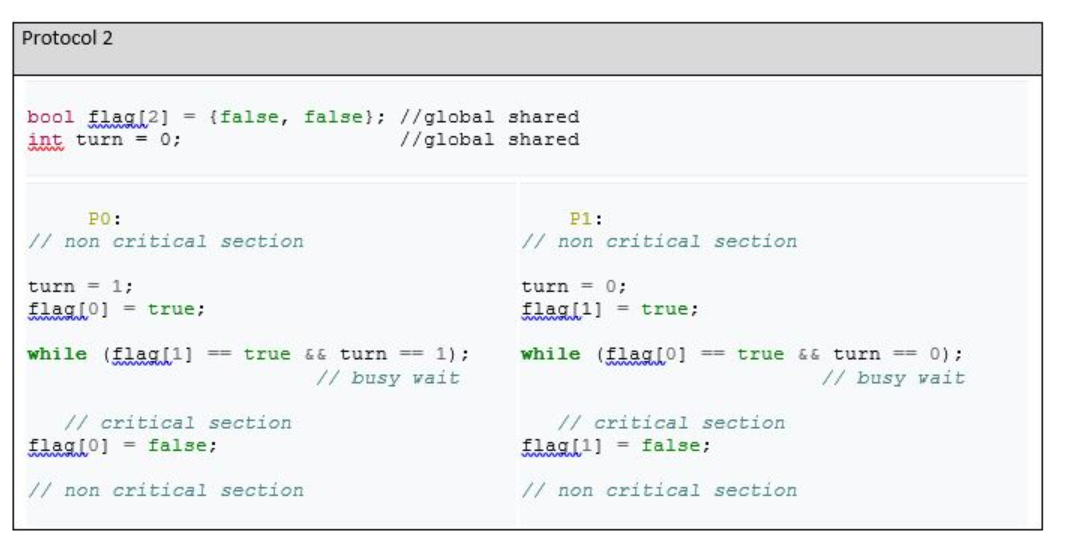


|  |
| --- |
| שני threads יכולים לשהות בו-זמנית ב- CRITICAL SECTION |
| threads עלולים להיכנס למצב DEADLOCK |
| הפרוטוקול מבטיח קדימות ל- thread שנוצר ראשון על פני ה- threads המתחרים |
| הפרוטוקול מבטיח קדימות ל- thread שנוצר אחרון על פני ה- threads המתחרים |
| הפרוטוקול פותר את בעיית ה- CRITICAL SECTION |

התכנית משתמשת במשתנים גלובליים משותפים הנגישים לשני ה- PROCESSES

**שאלה 7 (10 נקודות)**

נתונה התכנית הבאה (pseudo code) המיועדת ל  SYNCHRONIZATION בין שני PROCESSES - P0 and P1 הרצים במקביל. כל PROCESS מבצע את אותו פרוטוקול כניסה ויציאה ל/מ-  CRITICAL SECTION.

התכנית משתמשת במשתנים גלובליים משותפים הנגישים לשני  ה- PROCESSES. 

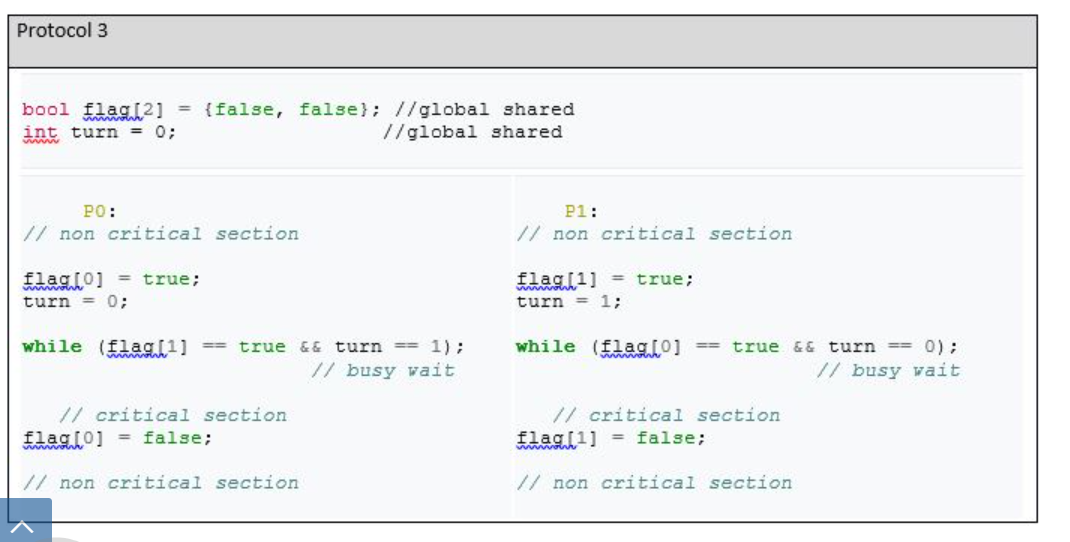
|  |
| --- |
| שני תהליכים יכולים לשהות בו-זמנית ב- CRITICAL SECTION |
| שני תהליכים עלולים להיכנס למצב DEADLOCK. |
| הפרוטוקול מבטיח קדימות לתהליך שנוצר ראשון על פני התהליכים המתחרים. |
| הפרוטוקול מבטיח קדימות לתהליך שנוצר אחרון על פני התהליכים המתחרים. |
| הפרוטוקול פותר את בעיית ה- CRITICAL SECTION. |

**שאלה 8 (10 נקודות)**

נתונה התכנית הבאה (pseudo code) המיועדת ל  SYNCHRONIZATION בין שני PROCESSES - P0 and P1 שרצים במקביל.

כל PROCESS מבצע את אותו פרוטוקול כניסה ויציאה ל/מ-  CRITICAL SECTION.

התכנית משתמשת במשתנים גלובליים משותפים הנגישים לשני  ה- .PROCESSES



|  |
| --- |
| שני תהליכים יכולים לשהות בו-זמנית ב- CRITICAL SECTION. |
| שני תהליכים עלולים להיכנס למצב  LIVELOCK. |
| הפרוטוקול מבטיח קדימות לתהליך שנוצר ראשון על פני התהליכים המתחרים. |
| הפרוטוקול מבטיח קדימות לתהליך שנוצר אחרון על פני התהליכים המתחרים. |
| הפרוטוקול פוטר את בעיית ה- CRITICAL SECTION. |

**שאלה 9 (10 נקודות)**

מערכת הקבצים של מערכת הפעלה מסוימת משתמשת בשיטת ה- I-node עם הפרמטרים הבאים:

• גודל הבלוק במערכת הקבצים הוא  1 Kbytes

• כתובת הבלוק היא 4 בתים (bytes)

• 10 שדות של ה I-node יכולים להחזיק ישירות את 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

חשבי מהו הגודל המקסימאלי של קובץ במערכת הקבצים הזו.

הערה:

לצורך החישוב

1 GByte = 1024 \* 1024 Kbyte   , 1 MByte = 1024 Kbyte

1 Kbyte = 1024 Byte

|  |
| --- |
| 778 KBytes |
| 16 MBytes |
| 16 GBytes |
| 17 GBytes |
| 32 MBytes |

שאלה 10 (5 נקודות)

איזה מהמשפטים הבאים נכון לגבי VMM (virtual machine monitor) בתצורת hosted ?

| **מודל תגובה** |
| --- |
| כל משאבי החומרה של המחשב (hardware resources) נשלטים באופן ישיר ע"י hypervisor . |
| ל- hypervisor חייב להיות מודול (MODULE) שמשתלב בתוך ה- kernel של מערכת ההפעלה המארחת. |
| רק הקצאות זיכרון ראשי עוברות דרך hypervisor , אבל פעולות קלט-פלט מבוצעות ישירות ע"י הOS - האורחת בלי שיתוף של  hypervisor . |
| בגלל שקיימת מערכת ההפעלה מארחת, יש אפשרות הרצה של יישומי משתמש הישר בתוך המכונה הוירטואלית בלי צורך בהתקנת מערכות הפעלה נוספות כלשהן בתוך שאר המכונה הוירטואליות. |
| כאשר יישום המשתמש מבצע את system calls ל-OS שרצה בתוך VMM , לא יהיה ניסיון מעבר  ל- kernel mode ואותה ה- OS לא תנסה לבצע את ה- system calls ב kernel mode . |
| שאלה 11 (5 נקודות)  טבלת הדפים של תהליך במערכת עם זיכרון וירטואלי נראית כדלהלן.  כל המספרים הם דצימליים, וכל הכתובות הן כתובות של בית בזיכרון (גודל מילה = לבית אחד).  גודל הדף הוא 2048 בתים.    לאיזו כתובת פיזית, אם יש כזו, תמופה הכתובת הוירטואלית 2078 ? ניתוח התשובות  | **מודל תגובה** | | --- | | 2078 | | 30 | | 4126 | | 54 | | 6174 | | [אין תשובה] | |

**שאלה 12 (5 נקודות)**

האם יש שגיאות בטבלת הדפים משאלה 11 ואם כן, מהן? סמני את התשובה המדויקת ביותר.

### ניתוח התשובות

| **מודל תגובה** |
| --- |
| אין שגיאות |
| יש רק שגיאה אחת - דף 5 נמצא (נוכח) במסגרת 2 שמספרה נמוך יותר ממספר המסגרת שדף 2 נמצא בה, הדבר אינו תקין |
| יש רק שגיאה אחת- דף 0 ודף 3 נמצאים (נוכחים) במסגרת עם אותו מספר, הדבר אינו תקין |
| יש רק שגיאה אחת - דף 2 ודף 4 נמצאים (נוכחים) במסגרת עם אותו מספר, הדבר אינו תקין |
| יש שתי שגיאות- דף 0 ודף 3 נמצאים (נוכחים) במסגרת עם אותו מספר, וגם דף 2 ודף 3 נמצאים (נוכחים) במסגרת עם אותו מספר, הדבר אינו תקין. |
| [אין תשובה] |

**שאלה 13  (10 נקודות)**

מערכת שרצה על מעבד עם 6 ליבות (CPU with 6 cores) כשכל ליבה יכולה להריץ THREAD אחד, מטפלת במקביל בהרבה משימות ובנויה בצורת boss-workers model עם העברת המשימות דרך התור.

כל המשימות משלבות פעולות CPU עם **פעולות קלט-פלט ביחס 50-50%.**  אין צורך להתחשב בזמן של CONTEXT SWITCH.

ה boss הוא חישובי בלבד (CPU  bounded 99%) והוא רץ עם עדיפות גבוהה בהרבה מ- workers .

כדי לשפר את מהירות המערכת, הותקן זיכרון ראשי גדול מאוד ולכן אין צורך בזיכרון וירטואלי.

הבדיקות הראו שהזמן הממוצע לביצוע משימה אחת (הזמן מהתחלת הביצוע ועד לסיומו) ע"י worker הוא 2 שניות.

זמן טיפול במשימה חדשה ע"י boss מרגע קבלת הבקשה ועד להכנסתה לתור המשימות הוא 0.3 שנייה.

קצב הגעת המשימות הוא 3 משימות בשנייה.

מהו מספר workers המינימאלי (אין ליצור workers מיותרים שלא תורמים משמעותית למהירות) שמומלץ ליצור כדי שזמן הטיפול במשימה (המתנה של משימה + ביצוע עד הסיום) יהיה הקצר ביותר ?

| **מודל תגובה** |
| --- |
| 5 |
| 9 |
| 12 |
| 7 |
| 10 |
| [אין תשובה] |

**שאלה 14 (5 נקודות)**

מה תהיה התשובה הנכונה על שאלה 13 כשמדובר על מעבד עם 3 ליבות (CPU with 3 cores) - כמה workers מומלץ ליצור במצב החדש כדי שזמן טיפול במשימה (המתנה + ביצוע עד הסיום) יהיה הקטן ביותר ?

### ניתוח התשובות

| **מודל תגובה** |
| --- |
| 6 |
| 4 |
| 3 |
| 5 |
| 2 |

**שאלה 15 (5 נקודות)**

איזה מהמשפטים הבאים נכונים כאשר משווים העברת מידע בין ה-PROCESSES דרך SHARED MEMORY להעברת מידע בין ה-PROCESSES דרך MESSAGE PASSING בעזרת שירות הודעות של מערכת הפעלה?

| **מודל תגובה** |
| --- |
| העברת מידע רב מהירה יותר דרך SHARED MEMORY מהעברת אותה כמות המידע דרך  MESSAGE PASSINGS , כשמדובר במספר רב של העברות. |
| לשתי הדרכים אותה מהירות בגלל ששתיהן מבוצעות ע"י מערכת ההפעלה והמהירות תלויה רק בה. |
| איזור SHARED MEMORY בכלל לא מיועד להעברת הנתונים, אלא רק לגישה למבני הנתונים המשותפים (קריאה בלבד). |
| MESSAGE PASSING אפשרית רק בין ה- THREADS השונים ולא בין ה- PROCESSES השונים. |
| כשמדובר ב- PROCESSES שונים, העברת מידע דרך SHARRED MEMORY בלתי אפשרית וחייבים להשתמש ב- MESSAGE PASSING להעברת מידע |
| [אין תשובה] |

**שאלה 16  (5 נקודות)**

לאיזה סוג שירותי ענן נכון לשייך שרות AWS (שירותי ענן של אמזון) שמאפשר להתחבר מרחוק דרך הדפדפן, לקבל גישה לטרמינל אמיתי של מערכת LINUX מסוג UBUNTU שרצה בענן שכוללת קומפיילר ועורך בסיסי, לבצע עליו פקודות מערכת הפעלה **ולהריץ** תכניות?

| **מודל תגובה** |
| --- |
| Infrastructure as a Service (IaaS)         בגלל שהמשתמש לא מודע לכמות המשאבים הפיזיים שהוא תופס. |
| Platform as a Service (PaaS)     בגלל שהעריכה מתבצעת בפלטפורמה שונה מהמחשב הלוקאלי. |
| Software as a Service (SaaS)     בגלל שהמשתמש מתחבר דרך דפדפן וכל האינטראקציה נעשית דרכו. |
| Platform as a Service (PaaS)     בגלל שהשירות כולל מערכת הפעלה והקומפיילר והעורך הם חלק ממנה. |
| Software as a Service (SaaS)     בגלל שהשירות כולל אפליקציות מוכנות בנוסף על חומרה ומערכת הפעלה |
|  |