

תראייל בית 3 - יבש 2

מבנה מערכות הפעלה

046209

מגישיים:

**דור משעלி 311510630
クリスティアン シュコル 208157826**

שאלה 1.

1. נתון הקטע קוד הבא:

א. (3 נק') מה יודפס למסך והאם ייתכן ויהיו שגיאות במהלך הרצאה? הסבירו.

```
jmp_buf buf1, buf2;  
  
void foo(int* a) {  
    int z = *a + 3; z=4  
    int *y = a; *y=1  
    int v = setjmp(buf1); V=0 V=1 V=2  
    if (v == 0) {  
        printf("A%d", z); A4  
        longjmp(buf1, 1);  
    } else if (v == 1) {  
        printf("B%d", *y); B1  
        longjmp(buf1, 2);  
    }  
}
```

מהתרגול למדנו כי `int setjmp(buffer)` שומרת את הרגיסטרים ב- `buffer` ומחזירה 0 . הפונקציה `void longjmp(buffer, value)` אינה חוזרת לעולם, משחררת את הרגיסטרים מ- `buffer` (כלומר לאירוע `setjmp` תחזיר `value` ולא אפס. התוכנית מתחילה ב `main` ובתחילתה יש קריאה לפונקציה `(buf1, setjmp)`, הפונקציה תחזיר 0 ולכן התנאי ב- `if` מתקיים.icut יודפס `C1`. בשורה הבאה נשלח לפונקציה `foo` את הכתובת למשתנה `z`. ערך `z` הוא 4. ערך המשתנה הולוקלי `z` הוא 0. נכנס לתנאי הראשון כי הוא מתקיים ואוז נדפס `A4`. נקפוץ ל `setjmp(buf1)` וכעת ערך המשתנה המוקמי בפונקציית `main` הוא ערך ה `value` שהוא 1. נכנס cutת לתנאי השני ונדפס `E2` . נגיע ל- `-` `longjmp(buf1,2)` ונקפוץ לאוותה שורה שהינו לפני כן כלומר `(buf1, setjmp)`,cut ערך ה `-` `value` הינו 2. התנאי השלייש מתקיים ואוז נדפס `F3`. נגיע לשורה לפני אחורונה בפונקציית `main` ונדפס `H4`. אין שגיאות בריצה והפלט יהיה לפי הסדר:

C1A4E2F3H4

```
int main() {
    int v = setjmp(buf1); V=0 / V=1 /V=2
    if (v == 0) {
        printf("C%d", ++v); C1 /V=1
        foo(&v);
        longjmp(buf1, 1);
        printf("D%d", ++v);
    } else if (v == 1) {
        printf("E%d", ++v); E2
        longjmp(buf1, 2);
    } else if (v == 2) {
        printf("F%d", ++v); F3
    } else if (v == 3) {
        printf("G%d", ++v);
    }
    printf("H%d", ++v); H4
    return 0;
}
```

ב. (4 נק') כתת מושגים ב-`jmp buf` יחד בלבד, כלומר כל מופיע של `buf2` הופך ל-`buf1.buf`. מה יודפס למסך והאם ייתכן ויהיו שגיאות במהלך הריצעה? הסבירו.

התוכנית מתחילה ב *main* ובתחילה יש קריאה לפונקציה *setjmp(buf1)*, הפונקציה תחזיר 0 ולכז בתנאי ב - *if* מתקיים. כתע יודפס *C1*. בשורה הבאה נשלח לפונקציה *foo* את הכתובת למשתנה *x*. ערך *x* הוא 4. ערך המשתנה הולוקלי *x* הוא 0 כיזה מה שהפונקציה *setjmp* תחזיר. נכנס לתנאי הראשון כי הוא מתקיים וזה נדפס *A4*. נ Kapoor ל *setjmp(buf1)* שנמצא בפונקציית *foo* כתע ערך המשתנה המקומי בפונקציית *main* הוא ערך ה *value* שהוא 1. נכנס כתע לתנאי השני ונדפס *B1*. נגיא ל - *longjmp(buf1,2)* ונ Kapoor לאוותה שורה שהיינו לפני כן כלומר *(buf1,1)*, *setjmp(buf1,2)*, כתע ערך ה *value* הינו 2. אף תנאי איינו מתקיים ואז נחזור לשורה אחורי שקרהנו לפונקציית *foo*. כתע, לא נדע לאםLK Kapoor כי *buf* יציבע למקום לא חוקי כי המחסנית נמחקה, ולכז נקבל *segmentation fault*. לכן, תהיה שגיאה ובגרובית דרבנן את השגיאה את הפלבו בהר.

C1A4B1

שאלה 2.

א. (3 נק') טענה: משותנה הנמצא במחסנית של חוט לא יctrיך הגנה ע"מ להבטחה
מניעה הדדית: הקפ' והסביר

נכון / לא נכון

לחותיטם יש גישה למחסניות של חוטים אחרים באותה תקופה ולכך למורות שהמשתנה נמצא במחסנית של חוט מסוים לא מבוטח שלא תהיה מניעה הגדית.

ב. (5 נק') האם ניתן deadlock בריצת החוטים? אם כן הראה סדר ריצה שגורם לdeadlock והצע שינויים ככל הצורך פשוט שמנוע את הdeadlock, אחרת הסבר מדוע כל ריצה מוגנת מבעיות סync'ロן?

בchan את קטעי הקוד הבאים של 3 חוטים ועננה על סעיף ב':
Thread 1:
1 lock(L1)
2 lock(L2)
3 // critical section requiring L1 and L2 locked
4 unlock(L2)
5 unlock(L1)

```
Thread 2:  
1 lock(L3)  
2 lock(L1)  
3 // critical section requiring L3 and L1 locked  
4 unlock(L1)  
5 unlock(L3)
```

```
Thread 3:  
1 lock(L2)  
2 lock(L3)  
3 // critical section requiring L2 and L3 locked  
4 unlock(L3)  
5 unlock(L2)
```

Thread 2:
1 lock(L1)
2 lock(L3)
3 //critical section requiring L3 and L1 locked
4 unlock(L1)
5 unlock(L3)

שאלה 3

טיפוס glass תומך בשתי פונקציות:
add(ingredient) •
פונקציית הוספה מרכיב לכו. יכול להיות vodka או orange ingredient.
.umbrella
serve_if_ready() •
פונקציה אוטומטית, המבצעת את סדר הפעולות הבא באופן אוטומי:
אם (כל שלושת המרכיבים נמצאים ב-g):
• הגש את g לסטודנט
• כוס חדשה g =
אחרת: <לא קורה כלום והתוכנית ממשיכה כרגיל>
* ל-(add() ו-serv_if_ready()) אין ערכי החזרה
* אין add()
* על כל כוס שימושת להחיל בדיקת מרכיב אחד מכל סוג

בפאב הטענו שלושה ברמנים וסוג קוקטיל יחיד בלבד - המרכיב השלישי מרכיבים: וודקה, תפוזים ומטריה.
בכל ערבות מגעים מספר סטודנטים, מזמינים את הקוקטיל ועל הברמנים להכין אותו מכל שלושת המרכיבים.
כל אחד מהברמנים תפקיד:
1. מזג וודקה
2. מוסיף תפוזים
3. שם מטריה
תמיד יש כוס אחת בלבד בהכנה להגשת ועל מנת שטעם הקוקטיל לא יתרס, הוודקה חייבת להימזג לפני שימושים נוספים (את המטריה ניתן להוסיף בכל שלב בהכנה הקוקטיל).
ניתן להניח כי פעולה ההגשת עצמה מתבצעת מיידית.
הניחו כי קיים המשנה המשותף g מטיפוס glass אשר מסמן את הכוס שבחינה:
glass g;

א. משוו פסאודו-קוד של הפונקציות עבור כל אחד מהברמנים (הניחו כי כל פונקציה נקראת בלא אינסופטי):

put_vodka()
put_orange()
put_umbrella()

Void Pat_Vodka (glass g)

down (add_lock)
g.add (vodka)
up (add_lock)
up (is_vodka)

g.serve_if_ready () {

up (vodka)
up (vodka)
down (orange)
down (umbrella)

Void Pat_umbrella (glass g) {

down (add_lock)
g.add (umbrella)
up (add_lock)

g.serve_if_ready () {

up (umbrella)
up (umbrella)
down (vodka)
down (orange)

Void Pat_orange (glass g) {

down (is_vodka)

init Technion_pub() {

Shared Semaphore is_vodka = 0

Shared Semaphore add_lock = 0

Shared Semaphore vodka = 0

Shared Semaphore orange = 0

Shared Semaphore umbrella = 0

g.serve_if_ready () {

up (orange)
up (orange)
down (umbrella)
down (vodka)



א. הנحو את ההגדרות הבאות:

```
typedef sem_t MyLock;
#define lock sem_wait
#define unlock sem_post
```

מה הבעיה שתיווצר? תארו רצף פעולות שעשוי לגרום לבעה זו ותארו על אילו תהליכי היא עשויה להשפיע.

ב' (2 נק)

א. הבעיה: *deadlock*

מרקחה אפשרי שלול לחובל לבעה: כניסה לפונקציה *recursiveFunction* גורם למשוך את המנעול *(lock(&LL))*, התנאי מתקיים ויש קריאה נוספת לפונקציה. התהילך נוהל שוב את ה- *lock(&LL)* הולך עד שהטהילך יהיה לפניו יחרר את המנעל. דבר שכמובן לא יקרה כי התהילך הנוכחי אמר לחסתיים.

: ה סוף / נובמבר 2019

ב. כדי לפחותה את הבעיה שציינית בסעיף א, אתם נדרשים למשוך (פסודו-קוד) מנגנון להשלימו את הקוד (במקומות הנΚודות) כדי שהמנעל החדש שלום יבוד נכוון: מותר להשתמש בסמפורים רגילים, תוך הנחה שהם אוטחלו ונגישים לכל התהליכים.

TypeDef struct {

... } MyLock;

Void init(MyLock* lck) {

... }

Void Lock(MyLock* lck) {

... }

Void Unlock(MyLock* lck) {

... }

שאלה 5

1. בשאלת זו נמשח מהותם החוטים (barrier). אופן השימוש בהם הוא: לאחר האתחול עברו מספר חוטים N דזוע מראש, חוט הקורה לפונקציה יצא להמתנה עד של N החוטים יקרוא ל- EnterBarrier. רק לאחר של החוטים "ונגסו למחסום", כלומר הפעילו את הפונקציה EnterBarrier, יוכל כל N החוטים המשיך את הביצוע שלהם.

א. (5 נק') מה הבעיה בקוד? הסבירו ותנו דוגמה לרכיבתה בעיתיה.

```
typedef struct{
    Int id_lock
    Int depth_rec
    Sem semaphore
} MyLock

Void Lock (MyLock* L){
    Int current_pid = getpid()
    If (current pid == L.id_lock)
    {
        L.depth_rec++
    }
    else{
        L.semaphore.down()
        L.id = current_pid
    }
    L.semaphore.up()
    L.id_lock = -1
}

Void UnLock(MyLock* *L){
    L.depth_rec--
    If(!L.depth_rec)
    {
        L.semaphore.up()
        L.id_lock = -1
    }
}
```

```
Void init(MyLock *L){
    L.id_lock = -1
    L.depth_rec = 0
    Sem_init(L.semaphore,1)
}
```

להלןימוש של Barrier:
בהתחלת החוט הראשי יקרא ל- :InitBarrier

```
1. int thread_count;
2. sem_t s1,s2;
3. int threads_inside;
4. void InitBarrier(int threads)
5. {
6.     thread_count = threads;
7.     threads_inside = 0;
8.     down(s2);
9. }
```

לפי הקוד אנחנו רואים שכארח החוט מספר N יכנס למחסום אז כל שאר החוטים יוכל למשוך בביצוע שליהם.

בקוד המתואר יכולת להיות בעיה כאשר מגעים שני החוטים האחרונים באותו זמן.

כארח הסטטוס נפתח נאמר שהחותוט 1(N) down הולכים לשין ומחייבים שיפתח הסטטוס אחד ואז הבעיה יכולה להיווצר בבדיקה בשלב זהה היא חוט N תחילה בפתיחת הקטע הקרקטי שלו לפניו שחותוט 1(N) עדכן את thread_inside ולכך גם עבור חוט N לא יעדכן בתנאי שורה 13 כך שבעצם לעולם לא יוכל להיכנס לתנאי ולא יפתח הסטטוס 2(N).

ב. (5 נק') ע"מ לתקן את הבעיה, ניסו לשנות את הקוד – שורות 19-23 הוחלפו בשורות הבאות:

במקרה זה אכן יקרה מצב שבו חוט N נכנס ללולאה. אם יווצר מצב שהחותוט N יעשה (up(2(N)) לפניו שחותוט 1(N) יעשה chown יוצר מצב בעצמו שחותוט N ישחרר בכל איטרציה את המנעל ולבסוף החוט 1(N) נכנס למןuel בלי יכולת לצאת ממנו. וכך יוצר מצב שתקוע חוט במנעל.

ג. עם שינוי זה ניתן להגיע deadlock. זאת ממש שחותוט הראשון נכנס למןuel 1(N) ככלומר יעשה down 1(N), הוא לא ענה על התנאי ולכןElse ששם יתקע בחown 2(N) ולא יפתח את 1(N). דבר זה יגרום לכך שכל החוטים יתקעו כבר בתחילת התיקוד כך שבעצם כל החוטים יתקעו והקוד לא יתקדם ונגיע deadlock.

ג. (3 נק') ע"מ לתקן את הבעיה, ניסו לשנות את הקוד שוב – שורות 19-23 הוחלפו הפעם בשורות הבאות:

```
19. else{
20.     threads_inside++;
21.     down(s2);
22.     up(s1);
23. }
```

האם הקוד תיקין עכשו? הסבירו