שמות המגישים:  
קווין סברסקי – 324589480

עמית שרמן --209284017  
<https://github.com/kevinsbarski/SPL--3>

**שאלה 1 חלק תיאוריטי**

**1.1** ה-**פוסט קונדישן** של המתודה זה שהמטבע נזרקה והתוצאה שהיא מחזירה. **הפרה-קונדישן** זה שהמטבע היא זמינה לפני שהמתודה נקראת.

**האינוריינטה** של המתודה היא השדה

Coin\_availbale א

Rounds\_counter ב

א צריך להיות פולס בזמן ההטלה

ב השדה צריך שיתווסף לו אחד בכל פעם שהמטבע הוטל

**1.2** בעיית בטיחות שיכולה להיגרם כתוצאה מהוספת המתודה הנל היא בעיית של **דדלוק**  בעיה של דדלוק קוראת כאשר שני תהליכונים מחכים אחד לשני שיעשו פעולה מסוימת אבל אף אחד מהם יכול להמשיך.

**דוגמה להמחשת הסיטואציה  
1)**שחקן מספר אחד מטיל עץ  
**2)** שחקן מספר אחד מחכה ששחקן מספר שתיים יטיל   
**3)**שחקן מספר שתיים מטיל עץ   
**4)**שחקן מספר שתיים מחכה ששחקן מספר אחד יטיל

**1.3**

**קטע הקוד הקריטי הוא** הוא כל קטע קטד אשר יש לו גישה למשאבים משותפים ויכול להשפיע של הנכונות והמיהמנות של התוכנית.

בתוכנית הזאת יש כמה קטעי קוד קריטיים כגון

**COIN\_AVAILBLE, ROUNDS\_COUNTER, GOOD\_FLIPS\_COUNTER,MAKE COIN AVALIBLE וכל קטע קוד אשר משותף למספר תהליכונים.**

**1.4**

במידה ונשתשמש בתנאי רגיל במקום בגארד סוספנשין הבעיה שעלולה להתעורר זה  **RACE CONDITION**וזאת משום שאם שני תהליכונים יבדקו אפ המטבע זמין בו זמנית הם יתחרו על לקחת את המטבע.

**1.5**

**אי אפשר** להיות בטוחים שרק תהליכון אחד יטיל מטבע בו זמנית בגלל שהשדה של הזמינות של המטבע הוא משמש כמנגנון סכרוני ששולט על הגישה עליו. אם מוחקים את השדה אי אפשר להיות בטוחים שרק תהליכון אחד יטיל את המטבע בו זמנית.

**שאלה 2 חלק תיאוריטי**

**2.1**

הסמפור בתכנית אינו מקיים מניעת הרעבה משום שלוגיקת המימוש שלו לא מתעדפת תהליכונים שכבר מחכים הרבה זמן על אילו שרק הרגע נקלטו.  
בשביל **בשביל** לשנות זאת ניתן לממש את המתודה שמחסירה 'כרטיס' בצורה הוגנת בעזרת מבנה נתונים של תור.  
**דרך נוספת** היא לייבא את  
**java.util.concurent.semaphore**   
הבנאי של הספריה הזאת יממש בצורה מובנת את ההוגנות של הסמפור.

**2.2**

כן נוכל לעשות זאת ע"י השימוש בו בכדי להגביל את מספר התהליכונים שניתן להם להפעיל את הקוד הקריטי באותו הזמן.  
כדי לעשות זאת ניתן להשתמש בהורדה בכדי לקחת 'כרטיס' לפני הכניסה לקטע הקריטי ובפעולת העלאה כדי לשחרר את ה'כרטיס' לאחר היצאה של הקטע הקוד הקריטי.



**שאלה 3 תיאורטי**

בעיית הבטיחות שעלולה להתעורר בקוד היא תנאי תחרות/קוד לא מיהמן

**1)תהליכון איי** קורא לפונקציית **ראן** של האוביקט **טראנסקשיין** שמנסה להעביר משאבים מ**פו-אקאונט** ל**בר-אקאונט.  
2)תהליכון איי** קורא ל**טריי טרנספר** שב**פו-אקאונט** שמנסה למשוך כמות מסוימת. **הוויטדרואו** נקרא ולוקח את המפתח של **פו-אקאונט.  
3)תהליכון בי** קורא ל**ראן**  שב**רנסקשיין**  שמנסה להעביר משאבים מ**בר-אקאונט** ל**פו-אקאונט.  
4)תהךיכון בי** קורא ל**טרי-טרנספר** ב**בר-אקאונט** שמנסה להעביר כמות מסוימת של משאבים. פונקציית ה-**וויטדרואו** נקראת ומנסה להשיג גישה למפתח שב**בר-אקאונט.  
5)תהליכון בי** נחסם כי המפתח ב**בר-אקאונט** לא חופשי.  
**6)תהליכון בי** מסיים את המשיכה מ**בר-אקאונט** וקורא למתודה **דפוסיט** שב**בר-אקאונט,**ומשיג מפתח  
 ל**בר-אקאונט.  
7)תהליכון בי** קורא ל**וויטדרואו** של **בר-אקאונט.** הוא קורא ל**דפוסיט** שב**פו אקאונט** ומשיג מפתח ל**פוו--  
 אקאונט.  
8)תהליכון איי וגם תהליכון בי** נמצאים אצלם מפתח ב**בנק-אקאונט ­שונה כל אחד, ושניהם עכשיו יכולים לשנות ולעדכן את החשבון של שני המשתמשים. זה יכול לגרום לחוסר מהימנות בקוד.**