קישור לגיט: https://github.com/YagelAtias/Project-2

1.

א) זה נגרם עקב תופעה שנקראת Thread-Context-Switch. משתתף 4 אכן הגיע ל100 מטר אבל זמן מעבד נלקח ממנו והועבר ל משתתף 3.

ב) בוודאי, כמו שנאמר בסעיף הקודם זמן מעבד נלקח למשתתף 3 לאחר שהוא סיים מרוץ, אבל לפני שהופעל סקריפט. ובהחלט יכול לגרום לכך שמשתתף (שהועבר לו הזמן מעבד) כן יסיים את המרוץ ויספיק גם להפעיל את הסקריפט. יותר מזה יכול גם לקרות מצב שתתקיים תופעה Race condition ושני משתתפים יזכו ב אותו מקום.

2.

הגדרת עדיפות (thread priority) אינה מבטיחה עדיפות ביצוע מוחלטת של thread. המכונה הווירטואלית של Java (JVM) יכולה לכבד את סדרי העדיפויות שנקבעו, אך היא אינה נדרשת לעקוב אחריהם בקפדנות. הסיבה לכך היא שניהול שרשור יכול להיות תלוי במערכת ההפעלה ובגורמים אחרים.

3.

ב Java יש שני שיטו, עיקריות לשימוש של תהלכון , בעזרת מחלקת Thread או במימוש ממשק Runnable. בשני המקרים, אנו יוצרים אובייקט THREAD ומפעילים אותם בשיטת start(). הקוד שברצוננו להפעיל במקביל חייב להיות ממוקם בשיטת run() של המחלקה Thread או Runnable.

היתרון לרשת את מחלקת THREAD שהוא קל בשימוש אבל אנחנו לא יכולים לרשת שום מחלקה אחרת.

היתרון בשימוש ממשק הוא שאנחנו יכולים לעבוד עם כמה ממשקים וגם לרשת מחלקה אחרת בו זמנית אבל בשביל להפעיל אותו אנחנו צריכים יותר מפעולה אחד.

אנו צריכים לבנות MyThread ולעטוף אתו את עם Thread בשביל שנוכל להפעיל את מתודת של ממשק Runnable.