

# הערות עבור תרגיל מעשי מס' 1

ציון התרגיל נקבע על בסיס:

- תקינות קוד (עם מתן דגש על מימוש נכון של התקשורת בין הלקוח לשרת עפ"י כל הכללים שנלמדו בכיתה)
- הנחות שלקחתם והאם הן תקינות – לדוגמה, הנחה שלקוח יכול לשמור קבצים בזכרון איננה תקינה.
- מעבר בדיקות אוטומטיות שכללו:
  - i. בדיקה בסיסית של שליחת מייל מלקוח אחד לאחר.
  - ii. בדיקה שכללה שליחת 10 מיילים בתרחיש הבא: לקוח A מתחבר, לקוח A שולח מייל ל B, A מתנתק, B מתחבר, קורא ומוחק - וכך חוזר 10 פעמים.
  - iii. שליחת מייל מלקוח אחד לאחר שכללה Attachment בגודל 300kb.
  - iv. פלט Valgrind תקין שמוודא התנהגות נכונה של התוכנית שלכם בזמן ריצה. שגיאות נפוצות שהיו עלולות לגרום לפלט Valgrind לא "נקי": אי שחרור זכרון, הסתמכות על משתנים לא מאותחלים, משאבים לא משוחררים בסוף התוכנית (בדגש על socket) ועוד גורמים אפשריים שניתן לראות בManual.

הערות כלליות הנוגעות לתרגיל:

- יש להיזהר בשימוש בפונקציות send ו-recv. מתוך ה-manual של send (ואנלוגי לחלוטין עבור recv):

```
Upon successful completion, the number of bytes which were sent is returned. Otherwise, -1 is returned and the global variable errno is set to indicate the error.
```

כלומר למעשה כתוב פה שכאשר send מחזירה מספר חיובי, אין זה מובטח שזה אותו המספר החיובי שהוא כמות הבתים שתכננתם לשלוח. רבים מכם לא בדקו את ערך ההחזרה של send או לחילופין בדקו אותו והשוו אותו ל -1 בלבד וזוהי שגיאה חמורה. הפתרון לבעיה הנ"ל היא שימוש ב `send_all` כפי שנלמדה בתרגול. נקודה עדינה עבור recv: בתרגיל הנוכחי רובכם מימשתם שרת שמטפל בלקוח אחד בכל רגע נתון בלולאה והדבר נכון עפ"י דרישות התרגיל. קיימים מקרים בהם אנשים שהשרת שלהם מטפל בלקוח אחד בלבד שגו והשתמשו ב `recv` במקום `receive_all` ולא בדקו ערכי החזרה בדומה למה שנכתב למעלה עבור `send`. שימו לב: בתרגיל הבא תידרשו לממש שרת שמטפל במספר לקוחות במקביל ושם `receive_all` לא יהיה פתרון נכון, כי הדבר עשוי לגרום ל"הרעבה" של לקוחות אחרים בזמן שאתם מצפים לקבלת כל הפקטות מההודעה הנוכחית של לקוח.

- שימו לב לסגנון התכנות שלכם. נקודות לציון:
  - i. רבים הגישו תרגילים שכללו שכפול קוד רב וכתובה של פונקציות ארוכות מדי ואלו דברים שבתרגיל זה לא הורדו עליהם נקודות, אבל אפשרי מאוד

שבתרגילים הבאים יורדו נקודות על סגנון לא טוב. כשממשים פרוטוקול תקשורת, נהוג להפריד את הקוד למודולים כך שהקוד המממש את הלוגיקה את ההתקשרות עצמה יהיה במודול חיצוני שניתן יהיה לקמפל עם הלקוח ועם השרת בנפרד, כך שיימנע שכפול קוד (כתיבת פונקציות זהות לחלוטין גם בקוד של הלקוח וגם בקוד של השרת).

ii. יש לשים לב שפונקציות לא יחרגו מגדלים "מוגזמים" ושם פונקציה כוללת בתוכה פונקציונליות רבה מדי, מפרידים אותה לתתי פונקציות מתאימות. דוגמה טובה היא מקרה בו השרת מספק מספר שירותים, אז כל שירות כזה יהווה פונקציה בפני עצמה.

iii. תשתמשו ב- Magic Numbers ותמנעו ככל האפשר ממספרים "סוררים" שנמצאים אצלכם בקוד. אם לדוגמה אתם רוצים שכל באפר שאתם שולחים יהיה בגודל 1024, יש להמנע מלכתוב 1024 ישירות בקוד, אלא לעטוף אותו במאקרו או בקבוע בשם `BUFF_SIZE`.

iv. נא לא להדפיס שורות Debug בקוד שאתם מגישים. אתם מוזמנים להדפיס הערות Debug לצרכי לוג כשאתם בודקים את התרגיל, אבל שימו לב שהקוד שמוגש הוא קוד שלא מדפיס אותן (או לחילופין תקמפלו את הקוד עם דגל - `DNDEBUG` תוך כדי הדפסה נכונה של הערות ה-Log). עפ"י רוב אין לי בעיה עם הודעות נוספות שאתם מחליטים להדפיס למען חוויית המשתמש, אך יש לשים לב שלא מדובר בהודעות לוג.

• קצת לגבי הנחות בתרגיל:

i. ניתנה לכם הנחה שהשרת יכול להחזיק את הקובץ בזכרון. ההנחה ניתנה כי שרתים בד"כ יותר "בנויים לתלפיות" מאשר מחשבי קצה ולא רצינו להכריח אתכם לממש איזשהו מנגנון שמירה על דיסק או דאטהבייס בצד שרת. לא אופשר לכם להניח שהלקוח יכול להחזיק קבצים שלמים בזכרון ולהיפר, נאמר לכם שיש לקרוא את הקובץ בחלקים ולשלוח כל חלק לשרת.

ii. שליחת attachments – למרות שניתנה דוגמה, חלקכם מימשתם את קבלת ה-attachments באופן שאותו לעיתים לא יכלתי לצפות בארגומנטים ל-`GET_ATTACHMENT`:

1. חלקכם נתנו נתיב מלא לקובץ ע"מ לשמור אותו (כולל שם הקובץ)
2. חלקכם הניח קבלת נתיב לתיקיה בלי '/' בסוף
3. חלקכם הניח קבלת נתיב לתיקיה עם '/' בסוף
4. ועוד...

עפ"י הוראות התרגיל ההנחה הנכונה היא 3 וזאת ההנחה שכולכם נדרשים לממש לתרגיל הבא.

בגלל שכנראה עפ"י רוב קרתה אי הבנה, אז במקרה שאתם **בטוחים בוודאות** שמנגנון שליחת הקבצים שלכם עובד והורדו לכם נקודות עליו, אתם מוזמנים לערער ולהגיד איך הנחתם שליחת קבצים.

.iii שימו לב לאופן בו אתם מתמודדים עם קלט. אם נאמר לכם בהגדרות התרגיל שהמשתמש מקיש:

User: <username>

Password: <password>

אז יש לאפשר לו להקיש כך. נסיונות להכין חלק מהשדות מראש (לדוגמה להדפיס על המסך User) מכשילים בדיקות אוטומטיות. יש להקפיד על כל ההנחיות הקשורות בקלט מהמשתמש כ-Case Sensitive ואין בהם מקום לאלתור, כנ"ל לגבי שמות קבצי ההרצה. אם ברצונכם להדפיס למסך הודעות אינפורמטיביות נוספות בזמן שהמשתמש מחובר (כמו שגיאות usage), אין מניעה.

בהצלחה בתרגילים הבאים,  
מיקי.