# תרגיל מעשי 1

# תיאור פרוטוקול התקשורות:

## סוג המשתנים בתוך הודעה:

1. **מספר**: תמיד יהיה בן ארבעה בתים.
2. **מחרוזות**: רצף של בתים (ללא תוספת שלnull בסופם), שקודם לכן יופיע מספר המייצג את אורך המחרוזת בבתים.

## רישא של הודעה:

כל הודעה מתחילה בשני מספרים בעל תפקיד קבוע:

1. מספר המייצג את **אורך ההודעה** העתידה לבוא. (כלומר, לא כולל את ארבעת הבתים המייצגים את אורך ההודעה).
2. מספר המייצג את **סוג ההודעה.**

## סוגי ההודעות ותיאורן:

### Login request

##### מידע בהודעה (לפי הסדר):

* מספר ההודעה הינו 1.
* מכיל 2 מחרוזות:
  + שם משתמש .
  + סיסמת משתמש.

##### תשובה להודעה:

הודעת General message.

### Show inbox

##### מידע בהודעה (לפי הסדר):

* מספר ההודעה הינו 2.

##### תשובה להודעה:

**בהצלחה**: הודעת Show inbox respond  
 **בכשלון**: הודעת General Message

### Show inbox respond

##### מידע בהודעה (לפי הסדר):

* מספר ההודעה הינו 3.
* מכיל רשימה של מחרוזות באורך לא ידוע כך שכל מחרוזת מכילה את הפרמטרים הבאים מופרדים ברווחים:
  + מספר המייל בצד השרת.
  + שולח ההודעה.
  + נושא ההודעה.

##### תשובה להודעה:

אין.

### Get mail

##### מידע בהודעה (לפי הסדר):

* מספר ההודעה הינו 4.
* מכיל מספר המייצג את מספר המייל בצד השרת (ראה Show inbox respond)

##### תשובה להודעה:

**בהצלחה**: הודעת Get mail respond  
 **בכשלון**: הודעת General Message

### Get mail respond

##### מידע בהודעה (לפי הסדר):

* מספר ההודעה הינו 5.
* מכיל 4 מחרוזות:
  + שם המשתמש ממנו נשלח המייל.
  + שמות המשתמשים אליהם נשלח המייל (מופרדים בפסיקים).
  + נושא המייל.
  + תוכן המייל.

##### תשובה להודעה:

אין.

### Delete mail

##### מידע בהודעה (לפי הסדר):

* מספר ההודעה הינו 6.
* מכיל מספר המייצג את מספר המייל בצד השרת (ראה [Show inbox respond](#_Show_inbox_respond))

##### תשובה להודעה:

הודעת General message.

### Compose mail

##### מידע בהודעה (לפי הסדר):

* מספר ההודעה הינו 7.
* מכיל 3 מחרוזות:
  + שמות המשתמשים אליהם ישלח המייל (מופרדים בפסיקים).
  + נושא המייל.
  + תוכן המייל.

##### תשובה להודעה:

הודעת General message.

### General message

##### מידע בהודעה (לפי הסדר):

* מספר ההודעה הינו 8.
* מכיל מספר המייצג את סטטוס הפעולה האחרונה שהשרת ביצע:

1. הפעולה עברה בהצלחה.
2. **תשובה לפעולת Login**: שם המשתמש לא ידוע.
3. **~~תשובה לפעולת Login~~**~~: הסיסמא לא נכונה.~~
4. המשתמש אינו מחובר ולכן אינו יכול לבצע פעולה.
5. **תשובה לפעולות Get/Delete mail**: מספר המייל לא ידוע.
6. **תשובה לפעולת Login**: המשתמש כבר מחובר לשרת.
7. שגיאה בצד השרת – נסה שוב לבצע את הבקשה.
8. שגיאה קשה בצד השרת שבעקבותיה הוא מפסיק לעבוד.
9. הבקשה לא חוקית – נסה שוב לבצע את הבקשה.
10. בעיה בחיבור ולכן החיבור ינותק מצד השרת.

##### תשובה להודעה:

אין.

### Welcome (ההודעה הראשונה שנשלחת: מהשרת אל הלקוח עוד לפני ביצוע Login)

##### מידע בהודעה (לפי הסדר):

* מספר ההודעה הינו 9.
* מכיל מחרוזת שמייצגת הודעת ברכה למשתמש שהתחבר לשרת.

##### תשובה להודעה:

אין.

### Quit (ההודעה האחרונה שנשלחת: הלקוח מודיע לשרת שהוא מסיים את ההתקשרות בינהם)

##### מידע בהודעה (לפי הסדר):

* מספר ההודעה הינו 10

##### תשובה להודעה:

אין.

# הרצת התוכנית:

כמתואר בדרישות התרגיל.

## שרת:

mail\_server users\_file [port]

## לקוח:

mail\_client [*hostname* [*port*][

# מבנה הקוד:

## תשתית:

מחלקות המשרתות את צד השרת והלקוח.

### Common - משותף

מכילה מידע כללי משותף גם לשרת וגם ללקוח, כגון: קידוד של ההודעות העוברות בינהם, פונקציות הדפסה למסך ועוד.

### Packet – משותף

מחלקה המשמשת לקריאה וכתיבה של מידע בצורה נוחה.   
היא עוטפת כתיבה וקריאה של מחרוזות ומספרים בצורה המתואמת ל[פרוטוקול שרת-לקוח שלנו](#_תיאור_פרוטוקול_התקשורות:).

### Socket - משותף

מחלקה העוטפת socket. מעבר למעטפה הרגילה של פונקציות הAPI, מכילה גם פונקציות כתיבה וקריאה (sendMessage, recvMessage) המשתמשות ב-[Packet](#_Packet_–_משותף) ומודעות ל[פרוטוקול שרת-לקוח שלנו](#_תיאור_פרוטוקול_התקשורות:) .

### FDSet – צד שרת

מחלקה שעוטפת את המבנה איתו עובדים בשימוש של פונקצית select.

### User – צד שרת

מחלקה המייצגת משתמש.

### MailObj – צד שרת

מחלקה המייצגת אימייל.

### Inbox- צד שרת

מחלקה המייצגת תיבת דואר של משתמש. מכילה רשימת המיילים שנשלחו למשתמש מסויים, (רשימה של [MailObj](#_MailObj_–_צד)), ומידע אודות המשתמש המשוייך אליה (אובייקט [User](#_User_–_צד)).

### ServerSessionSocket – צד שרת

מחלקה שמצייגת session שפעיל כרגע בין הלקוח ובין השרת.  
יורשת מSocket ומכילה בתוכה [Inbox](#_Inbox-_צד_שרת).

## Client

מחלקה המייצגת את הלקוח. מבצעת את תהליך התחברות לשרת, קבלת פקודות מהמשתמש, שליחתם לשרת ודיווח על תוצאות הפקודות.

## Server

מחלקה המייצגת את השרת. מבצעת את תהליך האתחול שלו, מאזינה לבקשות ומבצעת אותם.

# מקרים שלא הוגדרו

## צד לקוח

* כאשר יש שגיאה עד שלב הLogin (כולל) אין למשתמש הזדמנות נוספות להכניס נתונים והתוכנית מפסיקה.
* לאחר ההתחברות ישנם סיבות שונות שהתוכנית יכולה להפסיק בגינם, לדוגמא:
  + הודעת שגיאה חמורה מצד השרת שבעקבותיו הוא צריך לסגור את החיבור
  + בעיות בsocket של הלקוח
  + בעיות הקצאת זיכרון

במידה והשגיאה אינה חמורה, לדוגמא הכנסת פקודה לא ידועה מצד המשתמש, תוכנית הלקוח תמשיך ותאפשר למשתמש להכניס פקודות נוספות.  
**בכל מקרה של שגיאה תכתב הודעה מתאימה למשתמש.**

* עוד דברים....

## צד שרת

# 

### 