

## תרגיל בית מעשי מס' 1

להגשה עד 28.11.2011

שימו לב: באתר הקורס תוכלו למצוא מסמך עם הנחיות מפורטות לגבי הגשת תרגילים מעשיים. על כל התרגילים המעשיים שתגישו הסמסטר לעמוד בהנחיות הללו.

### מטרה

בתרגיל זה נממש שרת דוא"ל פשוט ונכתוב לקוח עבורו. השרת והלקוח ישוחחו בפרוטוקול אפליקציה אשר תגדירו בעצמכם.

בתרגיל מעשי מספר 2 נרחיב את האפליקציה והפרוטוקול ונסיף להם יכולות.

### השרת

השרת ישמור רשימת לקוחות, וישמור את המיילים הממתינים לכל לקוח.

אתחול השרת מתבצע בשורת הפקודה

```
mail_server users_file [port]
```

כש – users\_file הוא ה – path לקובץ טקסטואלי tab-delimited שמכיל שתי עמודות: שם משתמש והסיסמה המוגדרת למשתמש. למשל:

```
Moshe      1234
Yossi      password123
Esther     abcde
```

(בין שני השדות יש tab).

הפורט להקשבה הוא פרמטר אופציונלי עם ערך ברירת מחדל 6423.

לאחר מכן השרת ממתין ומשרת לקוחות. לשם פשטות, מרגע שהשרת מתחיל לרוץ הוא אינו מסיים את ריצתו. בכל רגע נתון הוא נדרש לטפל רק בלקוח אחד (ולכן מותר לכם להשתמש בפעולות חוסמות). מצד שני, הוא צריך להיות מוכן לשרת מספר רב של לקוחות לאורך חייו (כל עוד הם לא מנסים להתחבר באותו הזמן).

לשם פשטות, אתם יכולים להניח שהשרת נדרש לשמור מספר קטן יחסית של מיילים – attachments, ולכן אתם יכולים להחזיק הכל בזיכרון ולא נדרשים לשמור מידע באופן persistent (על הדיסק למשל).

כשלקוח חדש מתחבר לשרת השרת שולח לו ברכה כלשהי, שהיא שורת טקסט לבחירתכם. למשל:

```
Welcome! I am simple-mail-server.
```

רשתות תקשורת מחשבים, סמסטר א' 2011/12  
ביה"ס למדעי המחשב, אוניברסיטת ת"א

לאחר מכן הוא ממתין ללקוח שיכניס שם משתמש וסיסמה. אם הם נכונים, השרת מקבל את הלקוח וממתין לפקודות.

## הלקוח

אתחול הלקוח מתבצע בשורת הפקודה:

```
mail_client [hostname [port]]
```

כאשר hostname ו-port הם פרמטרים אופציונאליים. ערך ברירת המחדל הוא hostname = localhost, לא ניתן לספק port ללא hostname. הפרמטר hostname יכול להיות שם או כתובת IP. port = 6423.

אחרי שהלקוח מקבל את הודעת הברכה של השרת הוא קולט מהמשתמש שתי שורות קלט: שם משתמש וסיסמה אותן הוא שולח לשרת. הן מוכנסות בפורמט הבא:

```
User: Moshe
```

```
Password: 1234
```

בהנחה שהשרת אישר את המשתמש, יודפס למשתמש:

```
Connected to server
```

המשתמש יכול להכניס את הפקודות הבאות:

```
1. SHOW_INBOX
```

מציגה את המיילים הממתינים למשתמש: כל מייל הוא שורה בפורמט הבא:

```
mail_id sender "subject" num_attachments
```

כלומר: מספר סידורי של המייל (שישמש את המשתמש אח"כ כדי לבקש לקרוא את המייל הזה וכו'), שם השולח, הנושא, ומס' ה- attachments שמצורפים למייל. הנושא יהיה מוקף בגרשיים. למשל:

```
1 Yossi "Hi there!" 0
2 Yossi "Funny pictures" 2
3 Moshe "reports of the last year" 12
```

לשם פשטות, המספר הסידורי של המייל הראשון שהמשתמש יקבל הוא 1, של המייל השני הוא 2 וכן הלאה. המספר הזה לא משתנה אף פעם מרגע שנקבע - למשל, אם הלקוח קיבל מיילים שמספרם 1,2,3 ומבקש למחוק את מייל 2, אזי המיילים שישארו יהיו עדיין מספר 1,3 - אין צורך לשנות את המספרים שלהם ל-1,2. באופן זה המספר הסידורי מזהה את המייל באופן ח"ע בכל ההתחברויות של הלקוח לשרת. תוכלו להניח שמספר המיילים שמשתמש יכול לקבל לכל אורך חייו השרת מוגבל ל-32000.

```
2. GET_MAIL mail_id
```

מציגה את שם השולח, שמות הנמענים, הנושא, שמות ה- attachments והטקסט של המייל שמספרו mail\_id. אם יש יותר מנמען אחד או יותר מ- attachment אחד הם מופרדים בפסיקים. למשל:

רשתות תקשורת מחשבים, סמסטר א' 2011/12  
ביה"ס למדעי המחשב, אוניברסיטת ת"א

From: Yossi  
To: Moshe, Esther  
Subject: Funny pictures  
Attachments: funny1.jpg, funny2.jpg  
Text: How are you? Long time no see!

אם אין attachments, השורה הרלוונטית לא תכיל מידע. למשל:

Attachments:

תוכלו להניח שכל שאר השדות מכילים מידע.

3. GET\_ATTACHMENT mail\_id attachment\_num "path"

שומרת ל – path את attachment מס' attachment\_num במייל שמספרו mail\_id .path  
יהיה מוקף בגרשיים. תוכנת הלקוח מדפיסה למשתמש Attachment saved בסיום התהליך.

4. DELETE\_MAIL mail\_id

מוחקת את המייל שמספרו mail\_id מהשרת.

5. QUIT

מסיים את ריצתה של תוכנת הלקוח

6. COMPOSE

שולחת מייל חדש. אחרי פקודה זו המשתמש יכניס את פרטי המייל באופן הבא (ובדיוק בסדר הזה):  
נמען/ים, נושא, גוף המייל, attachment paths. אם יש יותר מנמען אחד או יותר מ – attachment אחד  
הם מופרדים בפסיקים. ה – path של כל attachment מוקף בגרשיים. למשל:

To: Moshe, Esther  
Subject: Funny pictures  
Attachments: "~/documents/funny1.jpg", "~/documents/funny2.jpg"  
Text: How are you? Long time no see!

לשם פשטות, תוכלו להניח שגוף המייל מורכב משורה אחת (בלי enters באמצע המייל).

אם אין attachments, השורה הרלוונטית לא תכיל מידע. למשל:

Attachments:

כל שאר השדות חייבים להכיל מידע.

אחרי שהמייל נשלח בהצלחה (כלומר, כל המידע הגיע לשרת כשורה) תוכנת הלקוח מדפיסה למשתמש  
.Mail sent

**דוגמת ריצה**

בצד השרת:

mail\_server ~/users.txt

עמוד 3 מתוך 5

רשתות תקשורת מחשבים, סמסטר א' 2011/12  
ביה"ס למדעי המחשב, אוניברסיטת ת"א

בצד הלקוח (בפונט **ירוק** – קלט מהמשתמש):

```
mail_client
Welcome! I am simple-mail-server.
User: Yossi
Password: password123
Connected to server
COMPOSE
To: Moshe
Subject: Hi there!
Attachments:
Text: I have just installed the new mail client!
Mail sent
COMPOSE
To: Moshe, Esther
Subject: Funny pictures
Attachments: "~/documents/funny1.jpg", "~/documents/funny2.jpg"
Text: How are you? Long time no see!
Mail sent
SHOW_INBOX
QUIT
```

(שימו לב שלמשתמש לא היו מיילים ממתנינים בתיבת הדואר ולכן לא הודפס דבר אחרי SHOW\_INBOX).

כעת מתחבר לקוח שני:

```
mail_client
Welcome! I am simple-mail-server.
User: Moshe
Password: 1234
Connected to server
SHOW_INBOX
1 Yossi "Hi there!" 0
2 Yossi "Funny pictures" 2
GET_MAIL 1
From: Yossi
To: Moshe
Subject: Hi there!
Attachments:
Text: I have just installed the new mail client!
DELETE_MAIL 1
GET_ATTACHMENT 2 1 "~/downloads/"
Attachment saved
QUIT
```

### דרישות התרגיל

ראשית, עליכם לתכנן פרוטוקול אפליקציה מתאים שיעבוד מעל TCP. לאחר מכן, ממשו אותו כפי שנלמד בתרגול. האפליקציה סינכרונית, כלומר אפשר להשתמש בפקודות חוסמות.

את פרוטוקול האפליקציה יש לתעד בבירור, באופן שיאפשר לכל אדם לממש לקוח או שרת ש"ידברו" עם התוכנת שהגשתם.

הדגש בבדיקת הקוד שתגישו ינתן כמובן למימוש התקשורת ברשת. על המימוש להיות יעיל וחבוסטי (robust). אל תשכחו לבדוק שגיאות בערכי חזרה מפונקציות ולטפל בהם בהתאם.

למקרה שתבחרו להגיש בסביבת nova: ייתכן שזוגות רבים יבדקו את הפתרון שלהם על nova בו-זמנית. לכן, מומלץ להשתמש בפורט שרת שאינו פורט ברירת המחדל בעת בדיקת הקוד.