ה א ו נ י ב ר ס י ט ה ה פ ת ו ח ה

20937

תכנות מערכות דפנסיבי

חוברת הקורס – קיץ 2024ג

כתב: רועי מימרן

יולי 2024 – סמסטר קיץ – תשפ"ד

פנימי – לא להפצה.

תוכן העניינים

אל הסטודנט והסטודנטית,

אני מקדם את פניך בברכה עם הצטרפותך אל הלומדים בקורס "תכנות מערכות דפנסיבי".
בחוברת זו תמצאו את הדרישות לקבלת נקודות זכות בקורס, לוח הזמנים ומטלות.

בשל מצב המלחמה בין ישראל לחמאס, ודחיית פתיחת סמסטר א', הוחלט לצמצם את היקף החומר בשנת הלימודים תשפ"ד ולפיכך יחידות 6 ו-7 בקורס הן נושאי בחירה ולא חובה. בהתאם לזה, היקף העבודה בממ"נים יהיה קטן מהרגיל, אם כי מספר המטלות לא השתנה.

לקורס קיים אתר באינטרנט בו תמצאו חומרי למידה נוספים, אותם מפרסם/מת מרכז/ת ההוראה.
בנוסף, האתר מהווה עבורכם ערוץ תקשורת עם צוות ההוראה ועם סטודנטים אחרים בקורס.
פרטים על למידה מתוקשבת ואתר הקורס, תמצאו באתר שה"ם בכתובת:
http://telem.openu.ac.il
מידע על שירותי ספרייה ומקורות מידע שהאוניברסיטה מעמידה לרשותכם, תמצאו באתר הספריה באינטרנט www.openu.ac.il/Library.

שעות הייעוץ שלי בכל יום ד', בשעות 11:00-12:30, בטלפון 09-7781270. פגישה רצוי לתאם מראש.
ניתן לפנות גם בדואר אלקטרוני: roymim@openu.ac.il
מילת התנצלות לסטודנטיות בקורס: פניות המופיעות בחומר הלימוד מנוסחות בלשון זכר - זהו למרבה הצער הנוהג המקובל. הפניות האלו מכוונות, כמובן, לכל קוראי החומר.

לתשומת לב הסטודנטים הלומדים בחו"ל:

למרות הריחוק הפיסי הגדול, נשתדל לשמור אתכם על קשרים הדוקים ולעמוד לרשותכם ככל האפשר.
הפרטים החיוניים על הקורס נכללים בחוברת הקורס וכן באתר הקורס.
מומלץ מאד להשתמש באתר הקורס ובכל אמצעי העזר שבו וכמובן לפנות אלינו במידת הצורך.

אני מאחל לך לימוד פורה ומהנה.

בברכה,

רועי מימרן
מרכז ההוראה בקורס

1. לוח זמנים ופעילויות (20937 /ג2024)

שבוע הלימוד	תאריכי שבוע הלימוד	יחידת הלימוד המומלצת	מפגשי ההנחיה*	תאריך אחרון למשלוח הממ"ן (למנחה)
1	19.7.2024-11.7.2024	יחידה 2-1	מפגש 1	
2	26.7.2024-21.7.2024	יחידה 2	מפגש 2-1	
3	2.8.2024-28.7.2024	יחידה 3-2	מפגש 3-2	מטלת אופל 01 1.8.2024
4	9.8.2024-4.8.2024	יחידה 3	מפגש 4-3	
5	16.8.2024-11.8.2024 (ג צום ט' באב)	יחידה 4	מפגש 5-4	ממ"ן 11 11.8.2024
6	23.8.2024-18.8.2024	יחידה 5-4	מפגש 6-5	ממ"ן 12 18.8.2024
7	30.8.2024-25.8.2024	יחידה 6-5	מפגש 6	
8	6.9.2024-1.9.2024	יחידה 6 (רשות)	מפגש 7-6	ממ"ן 13 1.9.2024
9	11.9.2024-8.9.2024	יחידה 7 (רשות)	מפגש 7	ממ"ן 14 12.9.2024 ממ"ן 15 14.10.20247

מועדי בחינות הגמר יפורסמו בנפרד

** התאריכים המדויקים של המפגשים הקבוצתיים מופיעים ב"לוח מפגשים ומנחים".*

2. תיאור המטלות

קרא היטב עמודים אלו לפני שתתחיל לענות על השאלות

פתרון המטלות הוא חלק בלתי נפרד מלימוד הקורס - הבנה מעמיקה של חומר הלימוד דורשת תרגול רב. המטלות תיבדקנה על-ידי המנחה ותוחזרנה לך בצירוף הערות המתייחסות לתשובות.

מבנה המטלות

כל מטלה מורכבת מכמה שאלות. בראש כל שאלה מצוין משקלה היחסי בקביעת ציון המטלה.
את הפתרונות למטלת אופל 01 עליך להגיש באמצעות סימון התשובה הנכונה ביותר מבין האפשרויות באתר הקורס.
את הפתרונות לממ"נים עליך להגיש בקובץ שיוגש באמצעות מערכת המטלות. במידה ויש יותר מקובץ אחד, יש לארוז את הקבצים בפורמט zip. בחלק מהמטלות תהיינה הנחיות ספציפיות לגבי אופן ההגשה. אם השאלה בממ"ן אינה ברורה לך, אל תהסס להתקשר אל אחד מהמנחים (בשעות הייעוץ הטלפוני שלו) לצורך קבלת הסבר.
המטלות מלוות את יחידות הלימוד בקורס. להלן פירוט המטלות והיחידות שאליהן מתייחסת כל מטלה. בחלק מהמטלות תופענה גם שאלות המתייחסות ליחידות קודמות, שכבר נלמדו.
במטלות התכנותיות בקורס, יש להגיש רק קוד שאתם כתבתם במו ידכם. ניתן להשתמש בחבילות תוכנה שהן חלק מהסטנדרט של השפה או אלה שנלמדו באופן ספציפי בקורס, וכן בקוד לדוגמא שמופיע באתר הקורס. עם זאת חל איסור להשתמש בקוד ממקורות באינטרנט או בתוכנות Generative AI כגון ChatGPT, כמו כן חל איסור לפרסם באתר אינטרנט או במאגר ציבורי כגון Github קוד לפתרון משימות מהקורס, או להגיש קוד ממקור כזה.

ניקוד המטלות

כל מטלה נקבע משקל; ניתן לצבור עד 30 נקודות. חובה להגיש 2 מטלות מתוך המטלות 01-14, ובנוסף את ממ"ן 15 – סה"כ 19 נקודות לפחות. רצוי להגיש הרבה ככל האפשר.

יש להגיש לפחות 2 מטלות מתוך המטלות 01-14 כדי לגשת לבחינה

להלן פירוט הניקוד לכל מטלה:

ממ"ן	ניקוד
01	2
11	3
12	4
13	3
14	4
15	14

3. התנאים לקבלת נקודות זכות בקורס

א. הגשת מטלות בסך כולל של 19 נקודות לפחות ובכללן ממ״ן 15 חובה.

ב. ציון של 50 לפחות בממ״ן 15 וציון 60 לפחות בבחינת הגמר.

ג. ציון סופי בקורס (שקלול כל מרכיביו) 60 לפחות.

לתשומת לבכם!

פתרון המטלות הוא מרכיב מרכזי בתהליך הלמידה, לכן מומלץ שתשתדלו להגיש מטלות רבות ככל האפשר.

כדי לעודדכם להגיש לבדיקה מספר רב של מטלות הנהגנו את ההקלה שלהלן:

בחישוב הציון הסופי נשקלל את כל המטלות שציוניהן גבוהים מציון משימת הגמר. ציוני מטלות אלה תורמים לשיפור הציון הסופי.
ליתר המטלות נתייחס במידת הצורך בלבד. מתוכן נבחר רק את הטובות ביותר עד להשלמת המינימום ההכרחי לעמידה בתנאי הגשת המטלות. משאר המטלות נתעלם.

זכרו! ציון סופי מחושב רק לסטודנטים שעברו את משימת הגמר בציון 50 ומעלה והגישו מטלות כנדרש באותו קורס.

מטלת מנחה (ממ"ן) 11

הקורס: תכנות מערכות דפנסיבי - 20937

חומר הלימוד למטלה: יחידה 2 – שפת C++

מספר השאלות:4 **משקל המטלה:** 3

סמסטר: 2024ג **מועד אחרון להגשה:** 11.8.2024

שימו לב:
את המטלה יש להגיש באמצעות מערכת המטלות המקוונת באתר הבית של הקורס **בלבד**
את התשובה יש להגיש בקובץ zip המכיל את הקבצים בהתאם למפורט בשאלות.

שאלה 1 (10%)

מה יודפס בהרצת הקוד הבא ולמה? הסבירו היטב את תשובתכם והתייחסו למנגנון הפולימורפיזם כפי שהוא ממומש בשפת C++.

```
#include <iostream>

class Foo
{
public:
   Foo() { baz(); }
   virtual void baz() { std::cout << "Foo::baz()" << std::endl; }
};

class Bar : public Foo
{
public:
   Bar() {}
   virtual void baz() { std::cout << "Bar::baz()" << std::endl; }
};

int main()
{
   Foo *pFoo = new Bar();
   delete pFoo;
   return 0;
}
```

את הפתרון יש להגיש בקובץ Word או PDF.

שאלה 2 (40%)

בשאלה זו (בלבד) אין להשתמש בספריית התבניות STL של C++, ויש לממש את האובייקט המבוקש בעצמכם.

א. (10 נקודות) בנו מחלקה בשם my_vector הכוללת וקטור או קואורדינטה במרחב תלת-מימדי, ובה 3 קואורדינטות x, y, z מסוג double שהן משתני private.

ב. (10 נקודות) ממשו את הפעולות הבאות בעזרת פונקציות / שיטות ציבוריות במחלקה:

(1) פונקציות בנאי המקבלות 3 קואורדינטות, או בנאי ברירת מחדל ללא פרמטרים המחזיר את וקטור האפס (ראשית הצירים), או בנאי העתקה המקבל וקטור קיים ומחזיר וקטור זהה.

(2) פונקציות המחזירות קואורדינטה: getX, getY, getZ.

(3) פונקציית הדפסה, המאפשרת להדפיס את תוכן הוקטור באופן הבא:

```
std::cout << vec1;
```

(14, -6, 9)

ג. (10 נקודות) ממשו אופרטורים של חיבור וחיסור על וקטורים, כך שניתן יהיה לכתוב vec1+vec2 או vec1-vec2 ולהציב את התוצאה בוקטור או להדפיסה.

ד. (10 נקודות) ממשו פעולות כפל בשני האופנים הבאים:

(1) כפל וקטור בסקלר: מקבל וקטור ומספר מטיפוס float/double, יוחזר וקטור שבו כל הקואורדינטות יוכפלו במספר הנתון.

(2) מכפלה פנימית: מקבלת שני וקטורים, מחזירה את המספר הממשי:

$$vec1 * vec2 = x_1x_2 + y_1y_2 + z_1z_2$$

דוגמה לתכנית ראשית שאמורה לפעול עם המחלקה הזו:

```
my_vec a(1, -4, 6);
my_vec b(0, -8 6);
my_vec z ();
std::cout << "a+b is" << a+b << std::endl;
z.setY(14);
std::cout << "8 * b is" << 8*b << std::endl;
std::cout << "but z * b is" << z*b << std::endl;
```

את התכנית שכתבת יש להגיש בקובץ my_vec.cpp, ניתן גם לצרף קובץ my_vec.h.

שאלה 3 (30%)

א. (20 נקודות) בנה מבנה נתונים שבו ניתן לאחסן וקטורים מהסוג שיצרת בשאלה 2, ולתת לכל וקטור שם מילולי שנקבע ע"י המשתמש (מחרוזת שמכילה אותיות ואולי ספרות ורווחים, מתחילה באות).

ב. (20 נקודות) כתוב תכנית שלדוגמה שבה בונים 10 וקטורים שונים תחת השמות:
David, Dana, Moshe, Vered, Mohammed, Yasmin, Ahmed, Lucy, Naftali, Ayelet.

ושומרים אותם תחת מבנה הנתונים שיצרת. לאחר מכן שלוף את הוקטור ששמו Vered והצג את ערכיו (התכנית כמובן צריכה להיות כתובה באופן גנרי, כך שתוכל לשלוף גם וקטור קיים בשם אחר).

את התכנית שיצרת בשאלה זו יש להגיש בקובץ all_vecs.cpp, ניתן גם לצרף קובץ all_vecs.h.

שאלה 4 (20%)

בנו תכנית ++C אשר קוראת קובץ בפורמט CSV., מפצלת כל שורה לשדות, וכותבת על המסך את סכום השדות המספריים בכל שורה ולאחר מכן סיכום של כל עמודה, כאשר:

(1) ניתן לייצר קבצי CSV למשל בעזרת תוכנת Excel. אין להשתמש בשום ספרייה יעודית לקריאת וטעינת קבצי CSV, אלא לבצע את זה בעזרת קוד שלכם.

(2) יש לכתוב את התכנית בעזרת ספריות ייעודיות לשפת ++C כגון iostream, fstream ולא בפונקציות שפת C כגון fopen. fgets.

(3) לצורך הפשטות נניח שהשדות בכל שורה מופרדות בפסיקים, והם מכילים אך ורק אותיות, ספרות ותווי רווח ולא סימנים אחרים. כאמור לצורך החישוב שהתכנית מבצעת, אין להתייחס לשדות שמכילים אותיות.

(4) בלשונית יחידה 2 תמצאו קבצי עזר לממ״ן, כולל קובץ קלט לדוגמה ודוגמת הרצה שלו.

(5) את התכנית יש להגיש בקובץ read_csv.cpp .

בהצלחה!

מטלת מנחה (ממ"ן) 12

הקורס: תכנות מערכות דפנסיבי - 20937

חומר הלימוד למטלה : יחידה 3 – חולשות אבטחה בשפת C++

מספר השאלות: 2 **משקל המטלה:** 4

סמסטר: 2024ג **מועד אחרון להגשה:** 18.8.2024

שימו לב:
את המטלה יש להגיש באמצעות מערכת המטלות המקוונת באתר הבית של הקורס **בלבד**
את התשובה יש להגיש בקבצים בהתאם למפורט בשאלות.
את ניתוחי החולשות יש לכתוב בהתאם לפורמט המופיע בספר הלימוד בפרק 4, תחת Documentation and analysis, הוא נמצא גם בלשונית יחידה 3 באתר הקורס.

שאלה 1 (20%)

בחברת האשראי "קשה", לקוח יכול להיות בעל קרדיט הנע בין 100- ל- 1000 שקלים.

לקראת החגים, החליטו בחברה לצאת במבצע קידום מכירות ולשלוח מתנה ללקוחות בעלי קרדיט הגדול מ- 750 שקלים.

להלן הקוד לבדיקה האם לקוח זכאי למתנה:

```
bool is_entitled_for_promotional_gift(int ID)
{
        unsigned int bound = 750;
        int credit = get_credit(ID);
        return (credit >= bound);
}
```

ליעל הסטודנטית, קרדיט מאוד נמוך בחברת האשראי. מה עליה לעשות כדי שתוכל לזכות במתנה המיוחלת?

א. מצאו את החולשה, הגדירו אותה והציעו דרך לתקוף את המערכת
ב. תקנו את הקוד כך שההתקפה לא תעבוד
ג. כתבו מסמך המתאר את החולשה, ההתקפה והתיקון.

הגשה: מסמך בפורמט pdf או word

שאלה 2 (80%)

לפניכם תוכנה המדפיסה ל- stdout את הקלט שלה. הקוד זמין גם בקובץ mmn02-q2.cpp באתר הקורס.

1. קמפלו את הקוד, הריצו אותו והבינו כיצד הוא עובד.
2. מצאו חולשה והשתמשו בה על מנת לקרוא לפונקציה unreachable. יש להציג קלט מתאים.
3. תקנו את הקוד כך שההתקפה לא תעבוד.
4. כתבו מסמך מחקר עם הסבר על החולשה, ההתקפה וההגנה.

```
#include <cstdlib>
#include <cstring>
#include <iostream>
#include <string>

////////////////////////////////////////////////////////////////////////////
///////////
//
// -- IMPORTANT! --
//
// for this exercise to run correctly do the following:
//
// a. Disable ASLR:
//            VS:    Configuration Properties->Linker->Advanced ->
"Randomized Base Address"
//            g++: disabled by default in gdb
//
// b. Set the target binary to x86
//            VS: Build -> Configuration Manager -> Active solution platform
-> X86
//            g++: -m32 flag   (if fails try: sudo apt-get install gcc-
multilib g++-multilib)
//
// c. Debug mode:
//            VS: Build -> Configuration Manager -> Active solution
configuration -> Debug
//            g++: -g3 flag   (maximal debug information)
//
////////////////////////////////////////////////////////////////////////////
///////////


#define PROGRAM_NAME "echoutil"
#define VERSION "1.0"

#define VERY_SECRET_PASSWORD "Cowabunga!"

class Handler
{
   virtual void unreachable()
   {
      printf("%s", VERY_SECRET_PASSWORD);
      exit(0);
   }

   virtual void helper(const char *str)
   {
      std::string s = "0" + std::string(str);
      unsigned int x = std::stoul(s, nullptr, 16);
      printf("%c", x);
   }
```

```
public:

   void interpret(const char* str)
   {
      helper(str);
   }
};


void usage(int status)
{
   fputs("Echo the STRING(s) to standard output\n"
      "\n"
      "\t-n   do not output the trailing newline\n"
      "\t-e   enable interpretation of backslash escapes\n"
      "\n"
      "\tIf - e is in effect, the following sequences are recognized : \n"
      "\t\t\\xHH     byte with hexadecimal value HH(1 to 2 digits)\n"
      , stdout);

   exit(status);
}

void handle_escape(const char* str)
{
   struct
   {
      char buffer[16] = { 0 };
      Handler h;
   } l;

   // copy only the characters after the escape char
   const char* s = str;
   char* p = l.buffer;
   s++;
   while (*s)
      *p++ = *s++;

   // handle different options
   switch (l.buffer[0])
   {
   case 'x':
      l.h.interpret(l.buffer);
      break;

   default:
      fputs(str, stdout);
   }
}

char* dupenv(const char* varname)
{
#if defined(_WIN32)
```

```
   char* buff = NULL;
   size_t cnt;
   if (_dupenv_s(&buff, &cnt, varname) != 0)
      return NULL;
   return buff;

#elif defined(__linux__)

   const char* s = getenv(varname);
   if (!s)
      return NULL;
   return strdup(s);

#endif
}
int main(int argc, char** argv)
{
   bool display_return = true;
   bool do_escape = false;

   char* env = dupenv("ECHOUTIL_OPT_ON");
   bool allow_options = env != NULL;
   free(env);

   if (allow_options && argc == 2)
   {
      if (strcmp(argv[1], "--help") == 0)
         usage(EXIT_SUCCESS);

      if (strcmp(argv[1], "--version") == 0)
      {
         fprintf(stdout, "%s version %s\n", PROGRAM_NAME, VERSION);
         exit(EXIT_SUCCESS);
      }
   }

   --argc;
   ++argv;

   if (allow_options)
   {
      while (argc > 0 && *argv[0] == '-')
      {
         const char* temp = argv[0] + 1;
         size_t i;
         for (i = 0; temp[i]; i++)
            switch (temp[i])
            {
            case 'e': case 'n':
               break;
            default:
               goto just_echo;
            }
         if (i == 0)
```

```
            goto just_echo;

         // options are valid
         while (*temp)
            switch (*temp++)
            {
            case 'e':
               do_escape = true;
               break;

            case 'n':
               display_return = false;
               break;
            }

         argc--;
         argv++;
      }
   }

just_echo:

   while (argc > 0)
   {
      const char* s = argv[0];

      if(do_escape && s[0] == '\\')
         handle_escape(s);
      else
         fputs(argv[0], stdout);

      argc--;
      argv++;
      if (argc > 0)
         putchar(' ');
   }

   if (display_return)
      putchar('\n');

   exit(EXIT_SUCCESS);
}
```

דגשים:

א. עבור תרגיל זה יש **לבטל את מנגנון ה- ASLR** ולבנות את הקוד ב- **32 סיביות (x86)**

ב. קמפלו את הקוד בקונפיגורצית **debug** ועשו שימוש בדבאגר (מספיק שההתקפה תעבוד עם דבאגר).

ג. עבודתכם תיבדק במ"ה לינוקס (Ubuntu), באמצעות g++ ולכן מומלץ לעבוד עם סביבה זו.

הגשה: קובץ עם הקוד המתוקן ומסמך מחקר בפורמט pdf או word. רצוי לחבר אותם ביחד לקובץ zip.

בהצלחה!

מטלת מנחה (ממ"ן) 13

הקורס: תכנות מערכות דפנסיבי - 20937

חומר הלימוד למטלה: יחידה 4 – שפת פייתון ומטא-תכנות

מספר השאלות: 3 **משקל המטלה:** 3

סמסטר: 2024ג **מועד אחרון להגשה:** 1.9.2024

הגשה: יש להגיש קובץ zip. המכיל את כל קבצי המטלה.

שאלה 1 (30%)

ענו על כל שאלה באמצעות קובץ תכנית בן עד 10 שורות, והגישו את הקובץ למערכת המטלות.
כמובן יש להימנע מהצבת ערכים מפורשים בפתרון ויש לפתור באמצעות עיבוד של נתוני הפתיחה.
בשאלה זו אין צורך להשתמש בפונקציות.

א. נניח שמקבלים רשימת מילים a_list. לכל מילה, צריך לזהות אם היא מתחילה באות b.
אם כן, צריך לשנות את המילה כך שתתחיל באות גדולה ואחריה אותיות קטנות, ואת המילים המתוקנות (ורק אותן) לשמור ברשימה b_list.
בנה תוכנית כך שהרשימה b_list מאותחלת כרשימה ריקה ומוסיפים אליה כל מילה מתאימה לאחר תיקונה. למשל, אם הרשימה a_list היא:

```
a_list = ["apple", "banana", "carrot", "black", "box"]
```

תתקבל התוצאה: b_list = ['Banana', 'Black', 'Box']

ב. האם ניתן לבנות שורת פייתון אחת שתבצע פעולה דומה ותייצר את הרשימה המבוקשת b_list? אם כן כתוב את השורה, אין חובה לאתחל את הרשימה בדרך כלשהי.

ג. נתונה שורה המכילה משפט, למשל:

```
line = 'This line contains words and some have the letter o'
```

יש לכתוב תכנית המפרקת את המשפט למילים, לזהות את אלה שמכילות את האות o ולהעביר אותן במלואן לאותיות גדולות. יש לכתוב את המילים כשהן מופרדות בפסיק ורווח. עבור הקלט בדוגמה למעלה, למשל, אמור להתקבל הפלט הבא:

```
CONTAINS, WORDS, SOME, O
```

שאלה 2 (10%)

ענה על השאלה באמצעות קובץ תכנית בן עד 30 שורות, והגש את הקובץ למערכת המטלות.

א. צור מחלקה בשם AppleBasket שהבנאי שלה מקבל שני ארגומנטים חיצוניים: מחרוזת המייצגת צבע, ומספר המייצג כמות. הבנאי צריך לאתחל שני משתני מופע: apple_color ו- apple_quantity. כתוב שיטה הנקראת increase המגדילה את הכמות באותו מופע ב-1 בכל הפעלה. כמו כן יש לכתוב שיטה בשם __str__ למחלקה זו המחזירה מחרוזת בפורמט:

"A basket of [כמות] [צבע] apples."

Example1: A basket of 4 red apples.

Example2: A basket of 50 blue apples.

מהתכנית הראשית יש ליצור שני מופעים ולהדפיס את תוכנם כך שיופיעו שתי הדוגמאות, בלי לקרוא מפורשות לאף שיטה פרט להפעלה אחת של increase לכל מופע.

ב. צור מחלקת ירושה בשם GreenAppleBasket לסל של תפוחים ירוקים, בנה בנאי למחלקה זו שמפעיל את בנאי מחלקת הבסיס עם צבע נתון Green.

את התכנית יש להגיש בקובץ fruit.py .

שאלה 3 (60%)

א. בשאלה זו יש לבנות את המחלקות הבאות, עם בנאי מתאים לכל מחלקה:

- משתמשים, להם יש שם ומקצוע.
- מחלקה יורשת לפי מקצועות שונים: מהנדסים, טכנאים, פוליטיקאים.
- כמו כן יש לבנות מחלקות לסוגי מהנדסים: מהנדסי חשמל, מהנדסי מחשבים, מהנדסי מכונות.
- התכנית תקבל קלט מהמשתמשים בזמן ריצה, שיאפשר לו להוסיף מחלקות נוספות לתכנית בזמן ריצה, תוך קבלת המידע הבא מהמשתמש: שם המחלקה החדש, שם משתנה חדש למחלקה, שם שיטה חדשה למחלקה והיא תיווצר בזמן ריצה (ניתן להגדיר משתנה עם ערך מספר שלם מקרי, ושיטה שמבצעת פקודת print אחת.
- המשתמש יוכל להגדיר שם של מחלקה אם, כך שהמחלקה החדשה תוגדר כירושה ממחלקה זו. יש לוודא שמחלקת האם אכן קיימת.
- לאחר יצירת המחלקה, יש להדפיס את תוכן השדה והמילון: newClass.__name__, newClass.__dict__.
- את התכנית יש להגיש בקובץ pros.py .
- דוגמא לריצת התכנית:

Please enter the name of new class: **Student**

Please enter name of base class (blank if none): **Politician**

Please enter name of new method for class Student: homework

Please enter name of new attribute for class Student: major

Class Student created with base class: Politician

Class __name__ is: Student

Class __dict__ is: ……

ב. כתבו תכנית המקבלת שם של קובץ פייתון המכיל מחלקה (לדוגמא, הקובץ fruit.py משאלה 2, יש להתייחס רק למחלקה הראשונה בקובץ), ושורת קוד בפייתון, מוסיפה לכל השיטות במחלקה את הקוד. נסו את התכנית על שורת הקוד print("Hello") והפעילו את שיטות המחלקה. ניתן לגזור את אופן הפעלת שיטות המחלקה מתוך הדוגמאות בתכנית

הראשית. את השינוי בתכנית יש לבצע בזכרון התכנית ולא לייצר קובץ חדש במחשב. את התכנית שלכם יש להגיש בקובץ meta.py .

דוגמה לריצת התכנית:

Enter python file name: **fruit.py**

Enter a python code: **print ("Added code!")**

Added code!

A basket of 4 red apples.

Added code!

A basket of 50 blue apples.

Added code!

הערה: זוהי דוגמה בלבד, ומספר השורות המדויק של Added code יכול להשתנות בהתאם למימוש.

בהצלחה!

מטלת מנחה (ממ"ן) 14

הקורס: תכנות מערכות דפנסיבי - 20937

חומר הלימוד למטלה: יחידה 5 - תקשורת

מספר השאלות: 2 **משקל המטלה:** 4

סמסטר: 2024ג **מועד אחרון להגשה:** 12.9.2024

בתרגיל זה נממש תוכנת **שרת** לגיבוי ואחזור קבצים ותוכנת **לקוח** שתעבוד מולו.
השרת יכתב בשפת C++ והלקוח בשפת python.

שרת (50%)

השרת יאפשר לכל לקוח לשלוח אליו קבצים לגיבוי ולשלוף את הקבצים האלו במועד מאוחר יותר.

מאפייני השרת:

א. השרת יתמוך בפרוטוקול חסר מצב (stateless)[1], כלומר, לא ישמור מידע בין בקשה לבקשה (כל בקשה עומדת בפני עצמה).
ב. השרת יתמוך בריבוי משתמשים ע"י תהליכונים (threads)

אופן הפעולה של השרת:

1. בלולאה אין סופית: ממתין לבקשות
2. בעת קבלת בקשה, יוצר thread חדש ומפענח את הבקשה לפי הפרוטוקול הנתון
3. ממשיך לפעול לפי הבקשה שהתקבלה:
   a. בקשה לשמירת קובץ לגיבוי:
   קבצים הנשלחים ע"י הלקוח ישמרו לתוך תיקיה יעודית של השרת, לכל משתמש תהיה תת-תיקיה ובתוכה הקבצים של אותו משתמש.
   לדוגמא: עבור לקוח מספר 1234 וקובץ בשם mmn14.pdf השרת ישמור את הקובץ בנתיב:
   c:\backupsvr\1234\mmn14.pdf
   b. בקשה למחיקת קובץ:
   מוחק את הקובץ הקיים.
   c. בקשה לרשימת הקבצים הקיימים:
   השרת יצור קובץ טקסט המכיל את רשימת הקבצים עבור לקוח זה.
   שם קובץ הטקסט יהיה אוסף תווים רנדומלי באורך 32 תווים (אותיות גדולות, קטנות באנגלית ומספרים)
   d. בקשה לאחזור קובץ:
   השרת ישלח כתשובה ללקוח את הקובץ המבוקש
4. אחרי הצלחה השרת יחזיר סטטוס הצלחה בהתאם לפרוטוקול
בכל מצב של שגיאה, השרת יחזיר סטטוס שגיאה בהתאם לפרוטוקול

[1] קראו כאן על פרוטוקול חסר מצב: https://en.wikipedia.org/wiki/Stateless_protocol

לקוח (50%)

הלקוח יעבוד מול השרת בהתאם לפרוטוקול.

בתחילת הריצה כל לקוח ייצר מספר אקראי ייחודי בגודל 4 בתים. מספר זה ישמש בכל הבקשות שיישלחו לשרת.

כתובת השרת והפורט יקראו <u>מתוך קובץ</u> בצורה הבאה:

- שם הקובץ: server.info
- מיקום הקובץ: באותה תיקיה של קובץ פייתון הראשי
- תוכן הקובץ: כתובת IP + נקודותיים + מספר פורט
  לדוגמא:
  127.0.0.1:1234

שמות הקבצים לגיבוי ואחזור יקראו מתוך קובץ בצורה הבאה:

- שם הקובץ: backup.info
- מיקום הקובץ: באותה תיקיה של קובץ פייתון הראשי
- תוכן הקובץ: שמות קבצים בלבד ללא נתיב (הקבצים יהיה באותה תיקיה של קובץ פייתון הראשי) .
  לדוגמא:
  mmn14.pdf
  terminator2.avi

כך תראה תיקיה לדוגמא:

```
C:\openu\mmn14>dir /b
mmn14client.py
backup.info
mmn14.pdf
server.info
terminator2.avi

C:\openu\mmn14>type server.info
127.0.0.1:1234
C:\openu\mmn14>type backup.info
mmn14.pdf
terminator2.avi

C:\openu\mmn14>
```

אופן פעולת הלקוח:

1. **יוצר מספר אקראי ייחודי בגודל 4 בתים**
2. **קורא את כתובת השרת והפורט מתוך קובץ server.info**
3. **קורא את שמות הקבצים לגיבוי מתוך קובץ backup.info**

4. שולח בקשה לשרת לקבל את רשימת הקבצים הקיימים בגיבוי
- שרת מחזיר תשובה, יש להציג על המסך את רשימת הקבצים או את הודעת השגיאה שהתקבלה
5. שולח בקשה לשרת לשמירת הקובץ **הראשון** המופיע ב- backup.info
- שרת מחזיר תשובה, יש להציג על המסך את התשובה שהתקבלה (כולל שם הקובץ)
6. שולח בקשה לשמירת הקובץ **השני** המופיע ב- backup.info
- הדפסה של תשובת השרת למסך
7. שולח בקשה לשרת לקבל את רשימת הקבצים הקיימים בגיבוי
- הדפסה של תשובת השרת למסך
8. שולח בקשה לאחזור הקובץ **הראשון** המופיע ב- backup.info
- הדפסה של תשובת השרת למסך ושמירת הקובץ על הדיסק (לצד קובץ פייתון, בשם tmp)
9. שולח בקשה למחיקת הקובץ **הראשון** המופיע ב- backup.info
- הדפסה של תשובת השרת למסך
10. שולח בקשה לאחזור הקובץ **הראשון** המופיע ב- backup.info
- הדפסה של תשובת השרת למסך
11. סיום ויציאה

פרוטוקול התקשורת

עליכם לממש את הפרוטוקול הנתון מעל TCP.

כל השדות המספריים חייבים להיות עם ערכים גדולים מאפס (**unsigned**) ומיוצגים כ- **little endian**

בקשה:

Request	שדה	גודל	משמעות
כותרת (Header)	user id	4 בתים	מייצג את המשתמש
	version	בית	מספר גירסת לקוח
	op	בית	קוד בקשה
	name_len	2 בתים	אורך שם הקובץ
	filename	משתנה	שם הקובץ (ascii) **לא כולל תו מסיים** (null terminated)
תוכן (payload)	size	4 בתים	גודל הקובץ שנשלח
	Payload	משתנה	תוכן הקובץ (בינארי!)

בקשות אפשריות:

Op	משמעות	הערות
100	שמירה של קובץ לגיבוי	כל השדות מלאים
200	בקשה לאחזור קובץ	שדות size ו- payload לא קיימים
201	בקשה למחיקת קובץ	שדות size ו- payload לא קיימים
202	בקשה לרשימת כל הקבצים של הלקוח	שדות name_len, filename, size, payload לא קיימים

תשובה:

Response	שדה	גודל	משמעות
כותרת (Header)	version	בית	מספר גירסת שרת
	status	2 בתים	סטטוס הבקשה
	name_len	2 בתים	אורך שם הקובץ
	filename	משתנה	שם הקובץ (ascii) **לא כולל תו מסיים** (null terminated)
תוכן (payload)	size	4 בתים	גודל הקובץ שנשלח
	Payload	משתנה	תוכן הקובץ (בינארי!)

תשובות אפשריות:

Status	משמעות	הערות
210	הצלחה	הקובץ נמצא ושוחזר. כל השדות מלאים
211	הצלחה	רשימת כל הקבצים חזרה ללקוח. כל השדות מלאים
212	הצלחה	גיבוי הצליח / מחיקת קובץ הצליחה. שדות payload, size לא קיימים
1001	שגיאה	קובץ לא קיים. שדה size ו- payload לא קיימים
1002	שגיאה	אין קבצים על השרת ללקוח זה. רק שדות version ו- status קיימים
1003	שגיאה	שגיאה כללית. בעיה עם השרת רק שדות version ו- status קיימים

זיכרו! הפרוטוקול הוא בינארי.

כך תיראה לדוגמא בקשה לגיבוי קובץ:

offset					
0	1234	1	100	9	mmn14.pdf
17	29189	25 50 44 46 2D 31 2E 36 ...			

שימו לב!

הפרוטוקול מחייב ולא ניתן לעשות בו שינויים. כפועל יוצא, כל שרת ולקוח המממשים את הפרוטוקול יכולים לעבוד אחד מול השני.

דגשים לקוד שרת:

1. ממשו את התוכנה לפי עקרונות תכנות מונחה עצמים
2. מומלץ (אבל לא חובה) לעשות שימוש בספריות STL
3. ניתן ורצוי להשתמש ביכולות C++11 (לדוגמא פונקציות מסוג למדה, שימוש ב- auto וכו'..).
4. למימוש התקשורת עשו שימוש ב- winsock או בספרית boost
5. שימו לב לייצוג ערכים בזיכרון כ- little-endian או big-endian
6. לקוח יכול לשלוח קובץ בגודל דינמי **גדול**. חשבו על הדרך הנכונה ביותר לקבל כמות מידע גדולה מהלקוח.
7. הקפידו על תיעוד של הקוד (comments)
8. תנו שמות משמעותיים למשתנים, פונקציות ומחלקות. המנעו ממספרי קסם!
9. **אבטחת מידע**
חישבו לאורך כל הדרך על אבטחת מידע. האם בדקתם את הקלט? איך נעשה שימוש בזיכרון דינמי? האם מתבצעת המרת טיפוסים (casting) וכו'..
האם ואיך אפשר לתקוף את השרת? האם השרת יכול לתקוף את הלקוח?

דגשים לקוד לקוח:

1. השתמשו בפייתון **גירסה 3.9** ומעלה
2. ממשו את התוכנה לפי עקרונות תכנות מונחה עצמים
3. עשו שימוש בספריות פייתון הסטנדרטיות
4. תוכלו להעזר בספריה **struct** על מנת לעבוד עם נתוני התקשורת בנוחות (בקשות/תשובות)
5. שימו לב לייצוג ערכים בזיכרון כ- little-endian או big-endian
6. השרת מאפשר קבלת קובץ בגודל דינמי **גדול**. חשבו על הדרך הנכונה ביותר לשלוח כמות מידע גדולה לשרת
7. הקפידו על תיעוד של הקוד (comments)
8. **אבטחת מידע**
האם תוכלו לתקף את השרת בצורה כלשהי? האם השרת יכול לתקוף את הלקוח?

הגשה:

1. **שרת**
   א. עליכם להגיש רק את קבצי הקוד (כלומר קבצי h. ו- cpp.).
   שימו לב! על התוכנית להתקמפל ולרוץ בצורה תקינה (ללא צורך בתוספות קבצים ללא קריסות)
   ב. עבודתכם תיבדק במערכת הפעלה חלונות, באמצעות Visual Studio ולכן מומלץ לעבוד עם סביבה זו.
2. **לקוח**
   א. עליכם להגיש רק את קבצי הקוד (כלומר קבצי py.).
   שימו לב! על התוכנית לרוץ בצורה תקינה (ללא צורך בתוספות קבצים, ללא קריסות)
   ב. יש לכלול פונקציה ראשית בשם **main**. פונקציה זו תהיה הפונקציה הראשית של תוכנית הלקוח והיא תעבוד לפי אופן פעולת הלקוח המוצג לעיל.
   טיפ:
   תוכלו להשתמש במנגנון הבא כדי לאפשר עבודה אינטראקטיבית **וגם** הרצה של הקוד

```
if __name__ == "__main__":
```

בהצלחה!

מטלת מנחה (ממ"ן) 15

הקורס: תכנות מערכות דפנסיבי - 20937

חומר הלימוד למטלה : יחידות 1-5 ; שאלת בונוס על פרק 7.

מספר השאלות: 2 + 1 בונוס **משקל המטלה:** 14 נקודות

סמסטר: 2024ג **מועד אחרון להגשה:** 14.10.2024

שאלה 1 (80%)

בתרגיל זה תממשו תוכנת שרת ולקוח המאפשרות ללקוחות להעביר קבצים באופן מוצפן מהמחשב שלהם לאחסון בשרת. השרת ייכתב בשפת Python ואילו הלקוח ייכתב בשפת C++.

חשוב!

קראו היטב את כל המטלה לפני תחילת העבודה. וודאו שאתם מבינים היטב את פרוטוקול התקשורת ואת המבנה של תוכנת השרת והלקוח.

ארכיטקטורה

ארכיטקטורת התוכנה מבוססת על שרת-לקוח. הלקוח יוצר קשר ביוזמתו עם השרת, מחליף איתו מפתחות הצפנה ולאחר מכן מעביר לו את הקובץ המבוקש בתקשורת מוצפנת. הלקוח מוודא שהשרת קיבל את הקובץ באופן תקין ע"י השוואת checksum בשני הצדדים, ובמידה ולא עבר באופן תקין, מנסה להעביר שוב (עד 3 נסיונות). בעמוד 3 מתואר תרשים הזרימה של המערכת.

שרת

תפקיד השרת לנהל את רשימת המשתמשים הרשומים לשירות ולאפשר להם להחליף ביניהם הודעות מסוגים שונים.

א. השרת יכתב בשפת python, הגרסה הקובעת לבדיקה היא 3.12.1.
ב. השרת יתמוך בריבוי משתמשים ע"י תהליכונים (threads) או ע"י selector.
ג. גרסת השרת תהיה **3** (גרסה זו מופיעה בהודעות תקשורת מטעם השרת).
ד. השרת יפעל עם חבילת הצפנה PyCryptodome, ופרט לכך עם חבילות סטנדרטיות הכלולות במפרש.

פורט

השרת יקרא את מספר הפורט מתוך קובץ טקסט בצורה הבאה :

- שם הקובץ : port.info
- מיקום הקובץ : באותה תיקיה של קבצי הקוד של השרת
- תוכן הקובץ : מספר פורט
  לדוגמא :
  1234

נתונים

השרת ישמור את נתוני הלקוחות והקבצים שנשמרו בזיכרון (RAM). כמו כן יחזיק תיקיה מקומית שתכלול את הקבצים שיתקבלו מלקוחות.

אופן פעולת השרת

1. קורא את הפורט מתוך הקובץ port.info. (אם הקובץ לא קיים, להוציא אזהרה ולעבוד על פורט ברירת מחדל 1256. לא להגיע לנפילה עם Traceback במידה והקובץ לא זמין.)
2. אם תבחרו לענות על שאלה 3 : השרת בודק את בסיס הנתונים, אם כבר קיים, וטוען נתוני לקוחות שנרשמו בהפעלות קודמות.
3. ממתין לבקשות מלקוחות בלולאה אין סופית.
4. בעת קבלת בקשה מפענח את הבקשה בהתאם לפרוטוקול :

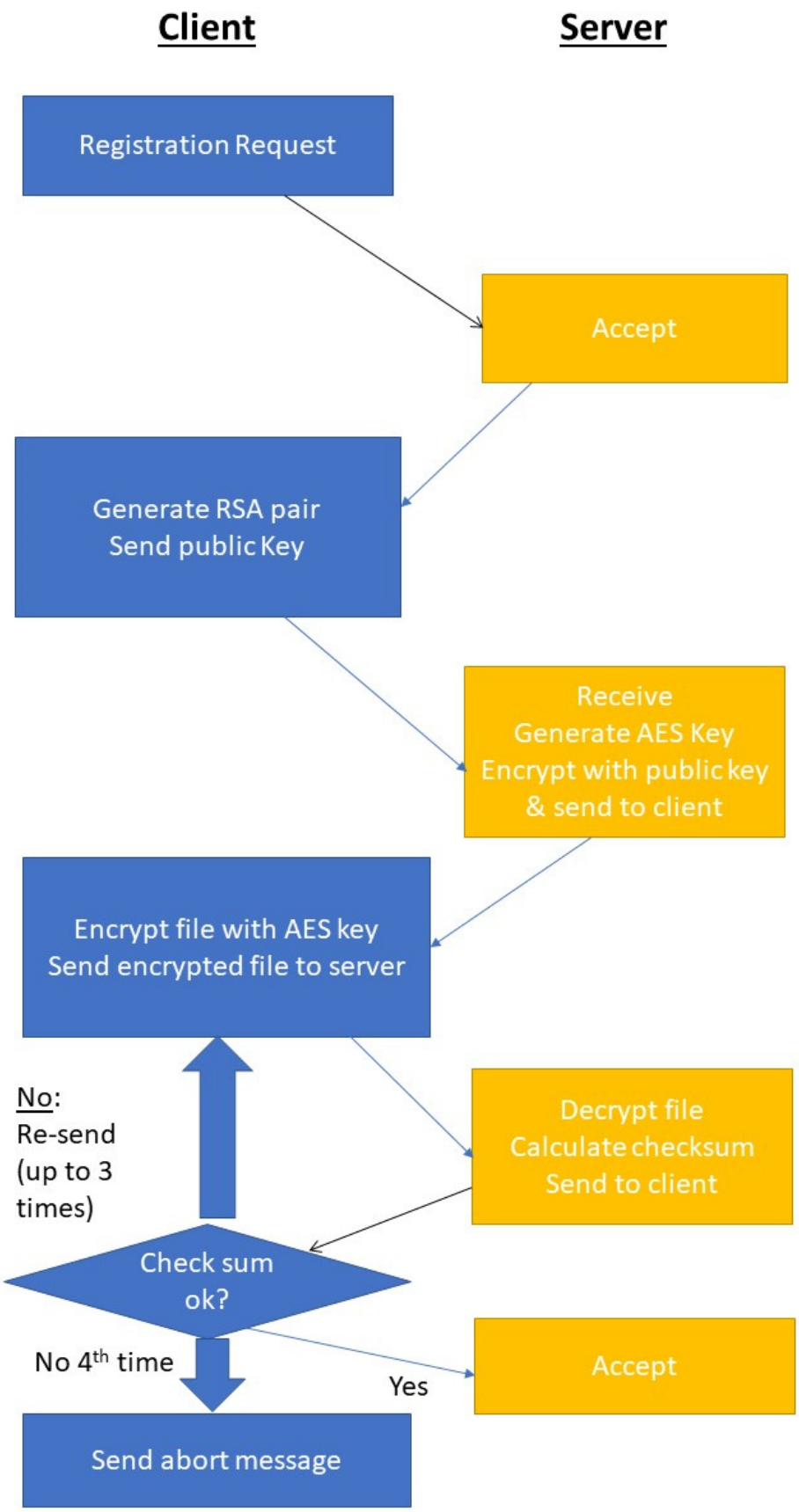
א. בקשה לרישום : במידה ושם המשתמש המבוקש כבר קיים, השרת יחזיר שגיאה. אחרת, השרת ייצר UUID חדש עבור המשתמש, ישמור את הנתונים בזיכרון ובבסיס הנתונים ויחזיר תשובת הצלחה.

ב. מפתח ציבורי מלקוח ייקלט ויעודכן בבסיס הנתונים. בתגובה, ייצור השרת מפתח AES, יצפין אותו בעזרת המפתח הציבורי וישלח בחזרה ללקוח.

ג. הודעה עם קובץ מוצפן : השרת יפענח את הקובץ המוצפן בעזרת מפתח ה-AES המקורי שנשלח לאותו לקוח, ויחשב את ה-CRC (שהוא הערך שמתקבל מפעולת checksum). **החישוב, בשרת ובלקוח, צריך להתבצע באופן זהה לפקודת cksum בלינוקס:**
https://www.howtoforge.com/linux-cksum-command/
בלשונית יחידה 7 תוכלו למצוא קוד לחישוב checksum ולהשתמש בו.

ד. השרת יקבל הודעת הצלחה מהלקוח (CRC אומת) או שליחה חוזרת של הקובץ עד 3 פעמים.

מצורף תרשים תהליך התקשורת העיקרי בין השרת ללקוח.


Client
Server
Registration Request
Accept
Generate RSA pair
Send public Key
Receive
Generate AES Key
Encrypt with public key
& send to client
Encrypt file with AES key
Send encrypted file to server
No:
Re-send
(up to 3
times)
Decrypt file
Calculate checksum
Send to client
Check sum
ok?
No 4th time
Yes
Accept
Send abort message

לקוח

תוכנת הלקוח תדע לתקשר מול שרת, להירשם (במידה ולא רשום מהפעלה קודמת), להחליף איתו מפתחות הצפנה ולאחר מכן להעביר אליו באופן מאובטח קובץ מהלקוח שיאוחסן בשרת. הלקוח אינו מתקשר או מודע ללקוחות אחרים במערכת.

א. תוכנת הלקוח תיכתב בשפת C++ תואמת גרסה 17, ותיבדק אצלנו בעזרת Visual Studio 2022.
ב. הלקוח יפעל על פי סדר פעולות קבוע, כך שניתן להפעילו במצב Batch mode.
ג. הלקוח יתבסס על הצפנה בעזרת חבילת CryptoPP.
ד. גרסת הלקוח תהיה 3.

קובץ הנחיות ללקוח

- שם הקובץ: transfer.info
- מיקום הקובץ: בתיקיה של קובץ ההרצה (.exe)
- תוכן הקובץ: שורה ראשונה – כתובת IP + נקודתיים + מספר פורט
- שורה שניה – שם הלקוח (מחרוזת עד 100 תוים)
- שורה שלישית – מסלול הקובץ לשליחה לשרת.
- <u>דוגמא</u>:

127.0.0.1:1234
Michael Jackson
New_product_spec.docx

שם ומזהה ייחודי[1]

הלקוח ישמור ויקרא את השם והמזהה הייחודי שלו מתוך קובץ טקסט בצורה הבאה:

- שם הקובץ: me.info
- מיקום הקובץ: בתיקיה של קובץ ההרצה (.exe)
- תוכן הקובץ:

שורה ראשונה: שם
שורה שניה: מזהה ייחודי בייצוג ASCII כאשר כל שני תווים מייצגים ערך hex בעל 8 סיביות.
שורה שלישית: מפתח פרטי שנוצר בריצה הראשונה של התוכנית בפורמט בסיס 64.
לדוגמא:

Michael Jackson 64f3f63985f04beb81a0e43321880182 MIGdMA0GCSqGSIb3DQEBA…

שגיאה מצד השרת

בכל מקרה של שגיאה הלקוח ידפיס למסך הודעה: "server responded with an error" וינסה לשלוח את ההודעה שוב, עד 3 פעמים, ואם עדיין לא יצליח, ייצא עם הודעת Fatal מפורטת.

[1] בתרגיל זה נעשה שימוש במזהה ייחודי גלובלי (UUID). לקריאה נוספת:
https://en.wikipedia.org/wiki/Universally_unique_identifier

<u>פעולות אפשריות</u>:

בקשת רישום

1. במידה והקובץ me.info לא קיים, הלקוח יקרא שם משתמש מהקובץ transfer.info וישלח בקשת רישום לשרת.
2. הלקוח ישמור בקובץ בשם **me.info** את השם והמזהה הייחודי שיקבל מהשרת.
3. במידה והקובץ כן קיים, הלקוח ישלח במקום זה לשרת בקשה להתחברות חוזרת. במצב כזה לא יוחלפו מפתחות RSA מחדש וייעשה שימוש במפתח הקודם מהקובץ priv.key.
שימו לב! במידה והקובץ כבר קיים הלקוח לא יירשם שנית.

מפתח ציבורי

הלקוח ייצר זוג מפתחות RSA, ציבורי ופרטי, ישלח את הציבורי לשרת וישמור את הפרטי בקובץ priv.key.

(בהתחברות חוזרת ייעשה שימוש חוזר בקובץ הזה ולא ייוצרו מפתחות RSA מחדש).

בתגובה השרת אמור לשלוח מפתח AES שהוצפן בעזרת המפתח הציבורי.

קבלת מפתח AES והצפנת הקובץ

לאחר שהלקוח מקבל את מפתח ה-AES, הוא פותח את המפתח בעזרת המפתח הפרטי של ה-RSA וקולט את מפתח ה-AES. בתגובה הוא מצפין בעזרתו את הקובץ שהוא נדרש להעביר, ושולח את הקובץ המוצפן לשרת. במקביל, הוא אמור לחשב את ה-CRC של הקובץ כדי שיוכל להשוות אותו ל-CRC שמתקבל מהשרת.

אימות השליחה בעזרת CRC

השרת אמור לקלוט את הקובץ המוצפן מהלקוח, לפתוח את ההצפנה בעזרת מפתח ה-AES, ולחשב גם הוא את ה-CRC ולשלוח אותו ללקוח לאימות.

פרוטוקול התקשורת

כללי

- הפרוטוקול הוא בינארי וממומש מעל TCP.
- כל השדות המספריים חייבים להיות עם ערכים גדולים מאפס (**unsigned**) ומיוצגים כ- **little endian**
- פרוטוקול זה תומך **בבקשות** לשרת **ותשובות** ללקוח. בקשות או תשובות יכולות להכיל "**הודעה**".
- הודעה עוברת בין לקוחות

זכרו! הפרוטוקול <u>מחייב</u> ולא ניתן לעשות בו שינויים. כפועל יוצא, כל שרת ולקוח המממשים את הפרוטוקול יכולים לעבוד אחד מול השני.

רישום למערכת

1. כל לקוח שמתחבר בפעם הראשונה נרשם בשירות עם שם (מחרוזת באורך מקסימלי של 255 בתים) ומעביר את המפתח הציבורי שלו
2. השרת יחזיר ללקוח מזהה ייחודי שנוצר עבורו או שגיאה אם השם כבר קיים בבסיס הנתונים.

פרטי הפרוטוקול

בקשות

מבנה בקשה מהלקוח לשרת. השרת יפענח את התוכן (payload) לפי קוד הבקשה.

בקשה לשרת

Request	שדה	גודל	משמעות
כותרת (Header)	Client ID	16 בתים (128 ביט)	מזהה ייחודי עבור כל לקוח
	Version	בית	מספר גירסת לקוח
	Code	2 בתים	קוד בקשה
	Payload size	4 בתים	גודל תוכן הבקשה
תוכן (payload)	payload	משתנה	תוכן הבקשה. משתנה בהתאם לבקשה

תוכן (payload)

התוכן משתנה בהתאם לבקשה. לכל בקשה מבנה שונה.

קוד בקשה 825 – רישום

שדה	גודל	משמעות
Name	255 בתים	מחרוזת ASCII המייצגת שם משתמש. **כולל תו מסיים!** (null terminated)

* **שימו לב:** השרת יתעלם מהשדה Client ID

קוד בקשה 826 – שליחת מפתח ציבורי

שדה	גודל	משמעות
Name	255 בתים	מחרוזת ASCII המייצגת שם משתמש. **כולל תו מסיים!** (null terminated)
Public Key	160 בתים	מפתח ציבורי של לקוח

קוד בקשה 827 – התחברות חוזרת (במידה והלקוח נרשם כבר בעבר)

שדה	גודל	משמעות
Name	255 בתים	מחרוזת ASCII המייצגת שם משתמש. **כולל תו מסיים!** (null terminated)

קוד בקשה 828 – שליחת קובץ

שדה	גודל	משמעות
Content Size	4 בתים	גודל הקובץ (לאחר הצפנה)

Orig File Size	4 בתים	גודל הקובץ המקורי (לפני הצפנה)
Packet number, total packets	4 בתים	2 בתים : מספר הודעה נוכחית 2 בתים : סה״כ מספר הודעות
File Name	255 בתים	שם הקובץ הנשלח
Message Content	משתנה	תוכן הקובץ. **מוצפן** ע״י מפתח סימטרי.

קוד בקשה 900 – CRC תקין

שדה	גודל	משמעות
File Name	255 בתים	שם הקובץ הנשלח

קוד בקשה 901 – CRC לא תקין, שולח שוב (לאחר מכן תגיע שוב בקשה 828)

שדה	גודל	משמעות
File Name	255 בתים	שם הקובץ הנשלח

קוד בקשה 902 – CRC לא תקין בפעם הרביעית, סיימתי

שדה	גודל	משמעות
File Name	255 בתים	שם הקובץ הנשלח

תשובות:

תשובה מהשרת

Response	שדה	גודל	משמעות
כותרת (Header)	Version	בית	מספר גירסת שרת
	Code	2 בתים	קוד התשובה
	Payload size	4 בתים	גודל תוכן התשובה
תוכן (payload)	payload	משתנה	תוכן התשובה. משתנה בהתאם לתשובה

קוד תשובה 1600 – רישום הצליח

שדה	גודל	משמעות
Client ID	16 בתים	מזהה ייחודי של לקוח

קוד תשובה 1601 – רישום נכשל

קוד תשובה 1602 – התקבל מפתח ציבורי ושולח מפתח AES מוצפן

שדה	גודל	משמעות
Client ID	16 בתים	מזהה ייחודי של לקוח
מפתח סימטרי מוצפן	משתנה	מפתח AES מוצפן ללקוח

קוד תשובה 1603 – קובץ התקבל תקין עם CRC:

שדה	גודל	משמעות
Client ID	16 בתים	מזהה ייחודי של הלקוח השולח
Content Size	4 בתים	גודל הקובץ (לאחר הצפנה)
File Name	255 בתים	שם הקובץ הנשלח
Cksum	4 בתים	CRC

קוד תשובה 1604 – מאשר קבלת הודעה, תודה.
(הודעה זו יכולה להתקבל כתגובה להודעה 900 או 902 מהלקוח).

שדה	גודל	משמעות
Client ID	16 בתים	מזהה ייחודי של לקוח

קוד תשובה 1605 – מאשר בקשה להתחברות חוזרת, שולח מפתח AES מוצפן – הטבלה זהה לקוד 1602:

שדה	גודל	משמעות
Client ID	16 בתים	מזהה ייחודי של לקוח
מפתח סימטרי מוצפן	משתנה	מפתח AES מוצפן ללקוח

קוד תשובה 1606 – בקשה להתחברות חוזרת נדחתה (הלקוח לא רשום או אין מפתח ציבורי תקין). במצב כזה על הלקוח להירשם מחדש כמו לקוח חדש.

שדה	גודל	משמעות
Client ID	16 בתים	מזהה ייחודי של לקוח

קוד תשובה 1607 – שגיאה כללית בשרת שלא טופלה באחד המקרים הקודמים (למשל נגמר המקום בדיסק, תקלה כללית בבסיס הנתונים ועוד).

הצפנה

פרוטוקול התקשורת משתמש בהצפנה סימטרית על מנת לקודד את הקבצים ובהצפנה אסימטרית על מנת להחליף מפתח בין סימטרי בין הלקוח והשרת.

בתרגיל זה השתמשו בצד הלקוח בספריה Crypto++[2] **(ראו דוגמת קוד ביחידה 7 באתר הקורס)**

הצפנה סימטרית

עבור הצפנה סימטרית השתמשו ב- AES-CBC.

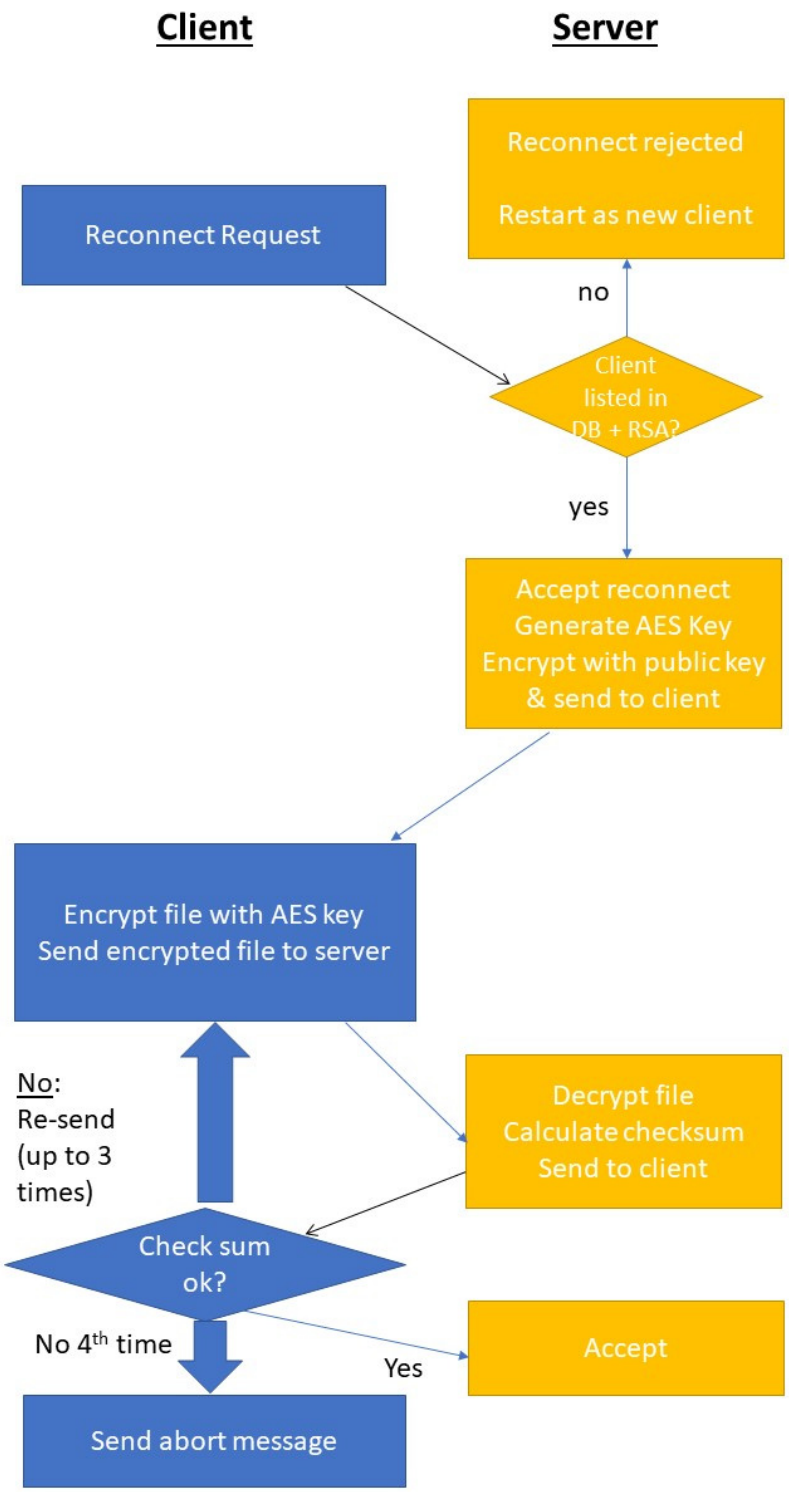
אורך המפתח **256 ביט**. ניתן להניח שה- IV מאופס תמיד (הזיכרון מלא באפסים).

שימוש כזה ב- IV לא בטוח אם משתמשים באותו מפתח בכל פעם, אך לצורך הממן הוא מספק.

הצפנה אסימטרית
עבור הצפנה אסימטרית השתמשו ב- RSA. אורך המפתחות **1024 ביט**.

[2] https://www.cryptopp.com/

<u>ניתוק והתאוששות</u>: התרשים הבא יתאר את התהליך:



שימו לב: הספריה Crypto++ מחזיקה מפתחות ציבוריים בפורמט X509[3]. פורמט זה מכיל Header לפני המפתח עצמו וערכים נוספים. לכן, גודלו הסופי (בצורה בינארית) הוא **160 בתים** (עבור מפתחות בגודל שונה גודלו הסופי של המפתח ישתנה בהתאם).

דגשים לפיתוח

1. מומלץ לעבוד עם מערכת לניהול קוד (כדוגמת גיט[4])
2. עבדו באופן מודולרי ובדקו את עצמכם כל הזמן
   א. זהו את המחלקות והפונקציות החשובות
   ב. **בצד השרת:**
   כיתבו קוד לטיפול בבקשה אחת. הוסיפו תמיכה בריבוי לקוחות בשלב מאוחר יותר
   ג. **בצד הלקוח:**
   ממשו את הרכיבים הגדולים באופן בלתי תלוי בחלקים אחרים של המערכת (תקשורת, הצפנה, פרוטוקול וכו').
3. ממשו קוד לבדיקה כבר בשלבים מוקדמים של הפרוייקט
   א. **בצד השרת:**
   השתמשו בהדפסות למסך או בכתיבה ללוג כדי לעקוב אחרי התקשורת. תוכלו גם לטעון את המודול לתוך ה- interpreter ולעבוד באופן דינמי.
   ב. **בצד הלקוח:**
   כיתבו פונקציות קטנות שבודקות חלקים נפרדים של המערכת. השתמשו בפונקציות הללו תוך כדי כתיבת הקוד עצמו.
4. כתיבת הקוד
   א. ממשו את התוכנה לפי עקרונות תכנות מונחה עצמים
   ב. שימו לב לייצוג ערכים בזיכרון כ- little-endian או big-endian
   ג. הקפידו על תיעוד של הקוד (comments)
   ד. תנו שמות משמעותיים למשתנים, פונקציות ומחלקות. המנעו ממספרי קסם!
   ה. הודעה יכולה להיות גדולה מאוד (בגודל דינמי). חשבו על הדרך הנכונה ביותר לקבל ולשלוח כמות מידע גדולה.
   ו. **אבטחת מידע** – חשבו לאורך כל הדרך על כתיבת קוד בטוח לפי העקרונות שלמדתם: האם בדקתם את הקלט? איך נעשה שימוש בזיכרון דינמי? האם מתבצעת המרת טיפוסים (casting) וכו'..
5. **לפני ההגשה**
   א. בדקו שהפרוייקט מתקמפל ורץ בצורה תקינה ללא קריסות או תלויות בספריות שונות (למעט הספריות הנדרשות לתרגיל)
   ב. מומלץ לייצר תיקיה חדשה ולהעתיק לשם את הקבצים המיועדים לשליחה. לייצר פרוייקט VS חדש, לקמפל ולהריץ
   ג. העבודה תבדק על מ"ה חלונות עם Visual Studio Community 2022 עם גרסת C++ 17

דגשים לקוד שרת:

1. השתמשו בפייתון **גירסה 3**
2. עשו שימוש בספריות פייתון הסטנדרטיות בלבד (פרט לספריית ההצפנה)!
3. תוכלו להעזר בספריה **struct** על מנת לעבוד עם נתוני התקשורת בנוחות

דגשים לקוד לקוח:

1. מומלץ (אבל לא חובה) לעשות שימוש בספריות STL

[3] https://en.wikipedia.org/wiki/X.509
[4] https://www.atlassian.com/git/tutorials/what-is-version-control

2. ניתן ורצוי להשתמש ביכולות C++11 ומעלה (לדוגמא פונקציות מסוג למדה, שימוש ב- auto וכו'..).
3. למימוש התקשורת עשו שימוש ב- winsock או בספרית boost

הגשה

שרת

1. עליכם להגיש רק את קבצי הקוד (כלומר קבצי py.).
שימו לב! על התוכנית להטען ולרוץ בצורה תקינה (ללא צורך בתוספות קבצים וללא קריסות).
2. יש לכלול פונקציה ראשית בשם **main**. פונקציה זו תהיה הפונקציה הראשית של תוכנית השרת והיא תעבוד לפי אופן פעולת השרת המפורט לעיל.
טיפ:
תוכלו להשתמש במנגנון הבא כדי לאפשר עבודה אינטראקטיבית **וגם** הרצה של הקוד

```
if __name__ == "__main__":
```

לקוח

1. עליכם להגיש רק את קבצי הקוד (כלומר קבצי h. ו- cpp.).
שימו לב! על התוכנית לרוץ בצורה תקינה (ללא צורך בתוספות קבצים, ללא קריסות)
2. עבודתכם תיבדק במערכת הפעלה חלונות, באמצעות Visual Studio ולכן מומלץ לעבוד עם סביבה זו.

וידאו עם דוגמת ריצה

עליכם להקליט וידאו ממסך המחשב, בו אתם פותחים שני חלונות cmd במקביל ומריצים את המערכת שפיתחתם. יש להפעיל קודם את השרת, לאחר מכן גם את לקוח, לעבור את התהליך של רישום לקוח והחלפת מפתחות כאשר ההודעות המתאימות מופיעות בשני החלונות במקביל, והעברת קובץ נתונים בינארי בגודל של כ-100 KB מהלקוח לשרת. בוידאו צריך להיות פרט מזהה הכולל את השם או תעודת הזהות שלכם, והוא צריך להימשך 2-5 דקות.

שאלה 2 (20%)

עליכם לנתח את הפרוטוקול המוצע בשאלה 1 ולמצוא בו חולשות פוטנציאליות. יש להגיש מסמך מחקר המפרט את החולשות שמצאתם, התקפות אפשריות והצעה לתיקון. בין היתר יש להציג טבלה בפורמט של מסמך החולשות מספר הלימוד, המופיע בלשונית יחידה 3.

שאלה 3 – בונוס (15%)

מוסיפים לשרת בסיס נתונים SQLite שיכלול טבלת רשימת המשתמשים, שמות מפתחות הצפנה שנשלחו להם, וטבלת רשימת הקבצים שהתקבלו מהם, והאם הקובץ עבר אימות מוצלח מול הלקוח בעזרת checksum. שמירת הנתונים תעשה ע"י טבלאות SQL בקובץ בשם defensive.db. היא תאפשר, במקרה של נפילה והתאוששות השרת, שליפת נתונים על לקוחות רשומים וקבצים שאוחסנו.

מידע על הלקוחות ישמר בטבלה בשם clients. מבנה הטבלה:

שם	סוג	הערות
ID	16 בתים (128 ביט)	מזהה ייחודי עבור כל לקוח. אינדקס
Name	מחרוזת (255 תוים)	מחרוזת ASCII המייצגת שם משתמש. **כולל תו מסיים!** (null terminated)
PublicKey	160 בתים	מפתח ציבורי של לקוח

LastSeen	תאריך ושעה	הזמן בו התקבלה בקשה אחרונה מלקוח
מפתח AES	256 ביט	מפתח AES שנשלח ללקוח

מידע על הקבצים שהתקבלו יישמר בטבלה בשם files. מבנה הטבלה:

שם	סוג	הערות
ID	16 בתים (128 ביט)	מזהה ייחודי עבור כל לקוח.
File Name	מחרוזת (255 בתים)	מחרוזת ASCII המייצגת שם קובץ כפי שנשלח מהמשתמש. **כולל תו מסיים!** (null terminated)
Path Name	מחרוזת (255 בתים)	מחרוזת ASCII המייצגת מסלול יחסי ושם קובץ כפי שמאוחסן בתיקיה שרת. **כולל תו מסיים!** (null terminated)
Verified	בוליאני	האם checksum אומת בהצלחה מול הלקוח

במקרה והשרת נפל, תהיה לו אפשרות לעלות מחדש, לטעון את בסיס הנתונים מקובץ Sqlite ולקוחות רשומים יוכלו לממש תהליך התאוששות מולו והמשך עבודה מבלי לשלוח קוד RSA מחדש.

הגשה

מסמך word או pdf .

הערה: את כלל קבצי המערכת יש לארוז לקובץ zip.