



מכון טכנולוגי חולון  
Holon Institute of Technology

המחלקה למדעי המחשב COMPUTER SCIENCE DEPARTMENT

## תכנות C

### מטלה 1 – לולאות ופונקציות

1. כתבו פונקציה **יעילה** בשם `multiplication` המקבלת שני מספרים **שלמים** (אשר יכולים להיות גם חיוביים, גם שליליים וגם אפס), מבצעת כפל שלהם על ידי שימוש בפעולות חיבור בלבד, ומחזירה את המכפלה.

לדוגמא:

עבור הפרמטרים, 4 ו- 15000 (הערך המוחזר יהיה 60000) הפונקציה תבצע 4 פעולות חיבור על-מנת לחבר את 15000 לעצמו, ולא 15000 פעולות חיבור שיחברו את 4 לעצמו.

גם עבור הפרמטרים, 4 ו- 15000 (התוצאה -60000), 4 ו- 15000 (התוצאה -15000) וכן 4 ו- 15000 (התוצאה -60000), הפונקציה תבצע 4 פעולות חיבור.

החתימה של הפונקציה `multiplication` היא:

```
int multiplication(int num1, int num2);
```

2. כתבו פונקציה בשם `primeNumbers` המקבלת שני מספרים **שלמים לא שליליים** `num1` ו- `num2` (`num2 > num1`) שמחשבת ומדפיסה את כל המספרים הראשוניים (המתחלקים רק בעצמו וב- 1) בטווח של `num1` עד `num2`.

לדוגמא:

עבור `num1 = 6`, `num2 = 13`, הפונקציה תדפיס את המספרים הבאים: 7, 11, 13.

חתימה של הפונקציה `primeNumbers` היא:

```
void primeNumbers(int num1, int num2);
```

יש להשתמש בפונקציית עזר בשם `prime` אשר מקבלת כפרמטר מספר **שלם לא שלילי** `num` ובודקת האם הוא ראשוני. אם המספר הוא ראשוני, הפונקציה תחזיר 1. אחרת עליה להחזיר 0. עבור `num = 1` ו- `num = 0` הפונקציה תחזיר 0. על הפונקציה להיות **יעילה ככל האפשר**.

חתימת הפונקציה תהיה `int prime(int num)`.

שימו לב: לכל מחלק `x` של `num` קיים משלים `y` אשר גם יהיה מחלק של `num` (`num = xy`). למשל,  $64 = 4 \cdot 16$  ואז 4 ו- 16 הם משלימים. לכן ניתן לחפש את המחלקים עד השורש הריבועי של `num`, במקום עד `num/2`. אם לא נמצאו ל- `num` מחלקים עד לשורש `num` אז אין לו גם מחלקים גדולים מהשורש.

3. כתבו פונקציה **יעילה** בשם `rotateNumber` המקבלת שני מספרים **שלמים לא שליליים**, `num` ו-`spins`. הפונקציה מבצעת מספר (ששווה לערך `spins`) הזזות מעגליות של ספרות ב-`num` כך שאחרי כל הזזה הספרה השמאלית ביותר הופכת להיות הימנית ביותר ושאר הספרות זזות עמדה אחת שמאלה. הפונקציה מחזירה את הערך שהתקבל.

#### לדוגמא:

עבור `number = 1234`, ו-`spins = 1` הפונקציה תחזיר 2341.  
עבור `number = 1234`, ו-`spins = 2` הפונקציה תחזיר 3412.  
עבור `number = 1000`, ו-`spins = 1` הפונקציה תחזיר 1.  
עבור `number = 6503`, ו-`spins = 2` הפונקציה תחזיר 365.  
עבור `number = 6503`, ו-`spins = 3` הפונקציה תחזיר 3650.  
עבור `number = 6503`, ו-`spins = 4` הפונקציה תחזיר 6503.  
עבור `number = 6503`, ו-`spins = 5000` הפונקציה תחזיר 6503.

החתימה של הפונקציה `rotateNumber` היא:

```
int rotateNumber(int number, int spins);
```

נקודה למחשבה: עבור שתי הדוגמאות האחרונות, האם יש צורך לבצע 4 ו-5000 סיבובים על המספר?

יש להשתמש בפונקציית עזר בשם `numberLength` אשר מקבלת מספר **שלם לא שלילי** ומחזירה את כמות הספרות שלו.

חתימת הפונקציה תהיה `int numberLength(int num)`.

#### הוראות:

- יש לאחד את כל השאלות הנ"ל בתוכנית אחת באמצעות תפריט הפונקציות `Ex1()`, `Ex2()`, `Ex3()` (ראו להלן) המשמשות להפעלת פונקציות השאלות 1, 2, 3 בהתאמה ומטפלות בקליטת הפרמטרים ובהדפסת התוצאות. יש להשתמש כתבנית בקובץ `Assignment_1_template.c` המצורף המכיל את הקוד (צריך לשתול בתוכו את הטקסטים של כל הפונקציות הנדרשות במקום המתאים).
- יש לבדוק את תקינות הקלט של מספרים לא שליליים באמצעות הפונקציה בשם `inputIntNonNegative` המשתמשת ב-`scanf` בתוכה. הפונקציה מחייבת לעשות קלט שוב ושוב כל עוד המספר הנקלט הוא שלילי ומחזירה את ערך המספר הנקלט שלא שלילי. חתימת הפונקציה תהיה `int inputIntNonNegative()`.
- אין צורך לבדוק אם המספר הנקלט הוא שלם. ניתן להניח שערך מוחלט של המספר הנקלט מספיק קטן כך שהוא מתקבל כ-`int`.
- אין צורך לבדוק בתוך פונקציה את תקינות הפרמטרים שלה.
- אין לכתוב פונקציות רקורסיביות.
- אסור להשתמש במערכים.
- יש להשתמש בשמות משמעותיים למשתנים ולתעד את הקוד עם הערות.

```

#define _CRT_SECURE_NO_WARNINGS
#define N 3

/*Libraries*/
#include <stdio.h>

int main()
{
    int select = 0, i, all_Ex_in_loop = 0;
    printf("Run menu once or cyclically?\n(Once - enter 0, cyclically -  

enter other number) ");
    if (scanf("%d", &all_Ex_in_loop) == 1)
        do
        {
            for (i = 1; i <= N; i++)
                printf("Ex%d--->%d\n", i, i);
            printf("EXIT-->0\n");
            do {
                select = 0;
                printf("please select 0-%d : ", N);
                scanf("%d", &select);
            } while ((select < 0) || (select > N));
            switch (select)
            {
                case 1: Ex1(); break;
                case 2: Ex2(); break;
                case 3: Ex3(); break;
            }
        } while (all_Ex_in_loop && select);
    return 0;
}

```