

## תכנות C

### מטלה 1 – לולאות ופונקציות

1. כתבו פונקציה **יעילה** בשם multiplication המקבלת שני מספרים **שלםים** (אשר יכולים להיות גם חיוביים, גם שליליים וגם אפס), מבצעת כפל שלהם על ידי שימוש בפעולות חיבור בלבד, ומהזירה 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 0), 4 ו- -15000 (התוצאה -60000), וכן -4 ו- -15000 (התוצאה 60000), הפונקציה תבצע **4 פעולות** חיבור.

חתימה של הפונקציה multiplication היא:

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

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

לדוגמא:

עבור  $13 = num1$  והפונקציה תדפיס את המספרים הבאים: 13, 11, 7.

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  $1 = num$  ו-  $0 = num$  הפונקציה תחזיר 0. על הפונקציה להיות **יעילה ככל האפשר**.

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

שימו לב: לכל מחלק x של num קיימים מושלים y אשר גם יריה מחלק של num ( $xy = num$ ). למשל,  $4 \cdot 16 = 64$  ואז 4 ו- 16 הם מושלים. לכן ניתן לחפש את המחלקים עד השורש הריבועי של num, במקומם עד  $\sqrt{num}$ . אם לא נמצא  $- num$  מחלקים עד לשורש num אז אין לו גם מחלקים גדולים מהשורש.

3. כתבו פונקציה **יעילה** בשם rotateNumber המקבלת שני מספרים **שלם** לא **שליליים**, **num** ו- **spins**. הפונקציה מבצעת מספר (שווה לערך **spins**) הזוזות מעגליות של ספרות ב- **num** כך שஅחרי כל הזזה הספרה השמאלית ביותר הופכת להיות הימנית ביותר ושאר הספרות זזות עד מה אחת שמאליה. הפונקציה מחזירה את הערך שהתקבל.

#### לדוגמא:

עבור  $1234$ ,  $number = 1234$ ,  $spins = 1$  הפונקציה תחזיר  $4123$ .  
עבור  $1234$ ,  $number = 1234$ ,  $spins = 2$  הפונקציה תחזיר  $3412$ .  
עבור  $1000$ ,  $number = 1000$ ,  $spins = 1$  הפונקציה תחזיר  $1$ .  
עבור  $6503$ ,  $spins = 2$ ,  $number = 6503$  הפונקציה תחזיר  $365$ .  
עבור  $6503$ ,  $spins = 3$ ,  $number = 6503$  הפונקציה תחזיר  $0365$ .  
עבור  $6503$ ,  $spins = 4$ ,  $number = 6503$  הפונקציה תחזיר  $3650$ .  
עבור  $6503$ ,  $spins = 5000$ ,  $number = 6503$  הפונקציה תחזיר  $6503$ .

חתימתה 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)`

#### הוՐאות:

1. יש לאחד את כל השאלות הנ"ל בתוכנית אחת באמצעות תפריט הפונקציות `(Ex1(), Ex2(), Ex3()`) (ראו להלן) המשמשות להפעלת פונקציות השאלות 1, 2, 3 בהתאם ומטפלות בקלט הפרמטרים ובהדפסת התוצאות. יש להשתמש כתבנית בקובץ `c Assignment_1_template` המצורף המכיל את הקוד (צריך לשתול בתוכו את הטקסטים של כל הפונקציות הנדרשות במקום המתאים).
2. יש לבדוק את תקינות הקלט של מספרים לא שליליים באמצעות הפונקציה בשם `scanf` ו המשמשת ב- `inputNonNegative`. הפונקציה מחיבת לעשוות קלט שוב ושוב כל עוד המספר הנקלט הוא שלילי ומחזירה את ערך המספר הנקלט שלא שלילי. חתימת הפונקציה תהיה `(int inputNonNegative())`.
3. אין צורך לבדוק אם המספר הנקלט הוא שלם. ניתן להניח שערך מוחלט של המספר הנקלט מספיק קטן כך שהוא מתקבל כ-`int`.
4. אין צורך לבדוק בתור פונקציה את תקינות הפרמטרים שלה.
5. אין לכתוב פונקציות רקורסיביות.
6. אסור להשתמש במערכות.
7. יש להשתמש בשמות שימושיים 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;
}
```