מבוא למדעי המחשב

הרצאה 3: מבוא לפונקציות

מבנה הזיכרון, חלוקת התוכנית
למספר פעולות לוגיות (פונקציות)

דקדוק בסיסי

סדר יום

- דיון ראשוני: מבוא לפונקציות.
- דוגמא ראשונה: דקדוק
- מושגים, למה צריך, איך כן, איך לא
- דיון: העברת פרמטרים, ערכים מוחזרים.
- דיון מסכם: מנגנון הקריאה לפונקציות, מחסנית, העברת פרמטרים.

מבוא לפונקציות:

<u>דיון</u>: ראשון פונקציה - תת תוכנית:

- דוגמאות
- קיימות: קלט, פלט, ערך מוחלט, שורש ריבועי, ...
- לא קיימות: האם מספר הוא ראשוני, ...

- תוכניות שונות יכולות להשתמש באותן הפונקציות!

מבוא לפונקציות דוגמא 1:

תוכנית לחישוב המספרים הראשוניים עד n:,

- חלוקה לתת משימות (פונקציות):
- פונקציה שמחשבת לכל מספר האם הוא ראשוני.
- פונקציה ראשית (main) שמשתמשת בפונקציה שהוגדרה כדי לחשב את כל המספרים הראשוניים עד n.

הערה: בדוגמא ראשונה זו נתעלם מנושא יעילות, ונכתוב את הקוד הנאיבי (והפשוט) ביותר:

מבוא לפונקציות דוגמא 1:

```
#include <iostream>
using namespace std;
int main()  {
      int n;
      cout << “Enter a maximal number: “;
      cin >> n;
      for(int x = 2 ; x < n ; ++x) {
          bool prime = isPrime(x);
          if (prime) cout <<x<<" ";
      }
      return 0;
}
```

מבוא לפונקציות דוגמא 1:

```
bool isPrime (int num)  {
    bool ans= true;
    for(int i=2; i<num; ++i)
        if (num%i==0) ans= false;
    return ans;
}
```

מבוא לפונקציות דוגמא 1:

```
bool isPrime (int num)  {
    for(int i=2; i<num; ++i)
        if (num%i==0) return false;
    return true;
}
```

מבוא לפונקציות דוגמא 1:

```
/** this function checks if n is a prime number.
    @param: num integer, checks if it is a prime number.
    @return: true if num is prime, else false. */
    bool isPrime (int num)  {
        for(int i=2; i<num; ++i)
            if (num%i==0) return false;
        return true;
    }  // isPrime function
```

מבוא לפונקציות דיון ראשון:

- מבנה התוכנית: קובץ יחיד, מספר פונקציות, main יחיד.
- כותרת פונקציה:

bool isPrime(int num) { … }

- ערך מוחזר, שם, פרמטרים (לפי סדר) {קוד לביצוע}
- מילים שמורת: return

מבוא לפונקציות מבנה התוכנית:

```
#include …

bool isPrime (int num)  {…}
…
int main() {…}
```

- קובץ יחיד (בינתיים).
- פונקציות רבות, main יחיד.

מבוא לפונקציות דיון שני:

- פונקציה: תת תוכנית המבצעת פעולה לוגית מסוימת (תיעוד).
- לפונקציה יכולים להיות פרמטרים, ערך מוחזר (שים לב למקבילה של תחום \ טווח)
- העברת פרמטרים:
- משתנים מקומיים, תחום הכרה.
- ערך מוחזר: (הפקודה return).
- קריאה לפונקציה.

מבוא לפונקציות הנדסת תוכנה

- ניתן כעת לחשוב בצורה אבסטרקטית,
- לתכנן את התוכנה ברמה של פונקציות (כותרת \ פעולה)
- פיתוח תוכנה – מספר מתכנתים.
- מחזור קוד.

פונקציות דיון רציני:

<u>מנגנון הקריאה לפונקציות:</u>

- למה להשתמש בפונקציות: מודולריות, חיסכון בקוד, תאימות, ...
- מנגנון קריאה לפונקציות:
  - הגדרה של מצב תוכנית: instruction pointer, זיכרון (טבלת משתנים).
  - מחסנית: סוג של זיכרון LIFO, משמש לשמירה של מצב התוכנית.

פונקציות דיון רציני (המשך):

<u>מנגנון הקריאה לפונקציות:</u>

- קריאה לפונקציה:
  - שמירה של מצב התוכנית.
  - קריאה לפונקציה העברת פרמטרים, פתיחה של סביבה חדשה במחסנית.
  - חזרה מפונקציה, ערך מוחזר, שחזור מצב התוכנית (pop).

דוגמא: מספרים אקראיים

אלגוריתמים רבים משתמשים במספרים אקראיים.
נרצה לתכנן מספר פונקציות לחישוב מספרים אקראיים:

- מה צריך?
- האם לכתוב בעצמנו (או להשתמש בקוד קיים)
- תכנון כללי

נתחיל בדוגמא פשוטה

דוגמא: מספרים אקראיים

```
#include <iostream>
#include <cstdlib>
using namespace std;
int main() {
    int d = rand();
    cout << "random number: "<<d;
    cout << "Max rand: "<<RAND_MAX;
    return 0;
}
```

דוגמא: מספרים אקראיים

```
#include <iostream>
#include <cstdlib> // for rand
using namespace std;
int main() {
    int d = rand(); // function call
    cout << "random number: "<<d;
    cout << "Max rand: "<<RAND_MAX;
    return 0;
}
```

דוגמא: מספרים אקראיים

- כעת נרצה לכתוב פונקציה שמחזירה ערך אקראי בין שני שלמים
- חתימה של הפונקציה?
- מגבלות:
  - פרמטרים
  - טווח ערך מוחזר
- תכנון מפורט?

דוגמא: מספרים אקראיים

```
/* returns a random number in the range [min,max).
    Note: uses srand seed to force a different seed each
    time. */
int rand_int_in_semi_open_range(int min, int max) {
    int dx = max-min;
    int ans = rand() % dx;
    ans = ans +min;
    return ans;
}
```

דוגמא: מספרים אקראיים

- המימוש הקודם סובל מבעיות של חוסר "יוניפורמיות"
- בסטנדרט של 2011 הוספה היכולת להחזיר מספרים שלמים רנדומיים לשפה.

דוגמא: מספרים אקראיים

בעיית ההרצה האקראית – גרעין קבוע:

נכתוב פונקציה שיש להריצה פעם יחידה בתחילת כל תוכנית – יוצרת גרעין 'אקראי' כל פעם.

```
#include <ctime>

…
void rand_seed() {
    srand((unsigned)time(0));
}
```

בדיקת פרמטרים – שגיאות אפשריות?

פונקציות סיכום ביניים:

- דקדוק בסיסי: כותרת, גוף הפונקציה ...
- מנגנון הקריאה לפונקציה:
- העברת פרמטרים , ערך מוחזר:
- דיון בהנדסת תוכנה

רוצים להדפיס "טבלת חזקות"

כלומר בשורה i וטור j יופיע המספר i^j

כאשר בניגוד לרב הקורס נתחיל לספור מאחד ולא מאפס

אסור להשתמש ב *

1	1	1	1
2	4	8	16
3	9	27	81