

შესავალი დაპროგრამებაში

ლექტორი:

სალომე ონიანი

ტელ. 571 39 40 22

ელ.-ფოსტა salome.oniani@iliauni.edu.ge



```
void max() {  
}
```

```
double max() {  
    return result;  
}
```

ფუნქციები

bool
char
int
float
double
void

```
int max() {  
    return result;  
}
```

```
bool max() {  
    return result;  
}
```

```
char max() {  
    return result;  
}
```

```
float max() {  
    return result;  
}
```

```
int sum( int a, int b) {  
  
    return a+b;  
}
```

```
void power() {  
  
}
```



დავწეროთ სტუდენტების რეგისტრაციის პროგრამა,
სადაც მომხმარებელი შეძლებს სტუდენტების
მონაცემების შეყვანას (მოიფიქრეთ მინიმუმ ხუთი
შესაყვანი პარამეტრი). სისტემას უნდა ჰქონდეს
შემდეგი მენიუ და ფუნქციები: 1. სისტემაში
სტუდენტის დამატება, 2. სისტემაში სტუდენტის
ძებნა, 3. სისტემიდან სტუდენტის წაშლა და 4.
სისტემაში დარეგისტრირებული სტუდენტების სიის
ნახვა

სასწავლო კვირა X

- პრეპროცესორის დირექტივები.
- ფაილში სხვა ფაილის ტექსტის ჩართვა.
- მაკროსები, პარამეტრიანი მაკროსები, მაკროსის გაუქმება, პირობითი კომპილაცია.

პრეპროცესორის დირექტივები

ყველაზე ხშირად გამოყენებადი პრეპროცესორის დირექტივებია

- #include

C-ზე დაწერილი ყველა მარტივი თუ რთული პროგრამული კოდი მოიცავს #include პრეპროცესორის დირექტივას

- #define

#include დირექტივა შეიძლება ჩაიწეროს ორი სახით

```
#include <ფაილის სახელი>
```

```
#include "ფაილის სახელი"
```

პრეპროცესორის დირექტივებს გავლენა არ აქვს პროგრამული კოდის მუშაობაზე, მაგრამ ისინი მონაწილეობენ პროგრამული კოდის კომპილაციაში

პრეპროცესორის დირექტივები

`#include` პრეპროცესორის დირექტივით შეგიძლიათ პროგრამულ კოდში შემოიტანოთ თქვენთვის სასურველი ფაილი ან ფაილები

- `h` გაფართოების ფაილები (ბიბლიოთეკები)
- `txt` გაფართოების ფაილები

ძირითადად `#include` დირექტივას იყენებენ `header` ფაილების პროგრამულ კოდში შემოსატანად, რომლიც წარმოადგენს შუალედურ ფაილებს და ინახება კომპიუტერის დისკზე.

იმ შემთხვევაში, როდესაც ვიყენებთ კომპილატორში ჩაშენებულ ფუნქციებს, როგორც არის მაგალითად `printf()` მისი შესაბამისი გამშვები ფაილი სასურველია ჩავწეროთ `#include <ფაილის სახელი>` ფორმატში

h გაფართოების ფაილები

ყველა კომპილატორში ჩაშენებულ ფუნქციას აქვს თავისი გამშვები ფაილი, სადაც აღწერილია ჩაშენებული ფუნქციის შესასრულებელი მოქმედებები

მაგალითად, ყველა ჩვენს მიერ დაწერილი პროგრამული კოდი მოიცავდა `#include <stdio.h>` იმიტომ რომ ყველა კოდში ვიყენებთ `printf()` ან/და `scanf()`. ანუ `printf()` და `scanf()` ჩაშენებული ფუნქციების გამშვები ფაილია `stdio.h`

h გაფართოების ფაილები

იმ შემთხვევაში, როდესაც ვიყენებთ ჩვენს მიერ შექმნილ გამშვებ ფაილს სასურველია გამოვიყენოთ `#include` "**ფაილის სახელი**" ფორმატი, რადგან როდესაც დირექტივის ასეთ ფორმას იყენებთ კოდში, კომპილატორი კომილაციის დროს ჯერ გამშვებ ფაილს ეძებს იქ სადაც თქვენი პროგრამული კოდის პროექტი ინახება და შემდეგ ჩაშენებული გამშვები ფაილების საქაღალდეში. ასეთი ძებნის თანმიმდევრობის გამო თქვენ შეგიძლიათ თქვენს მიერ შექმნილ გამშვებ ფაილს დაარქვათ უკვე კომპილატორში ჩაშენებული გამშვები ფაილის სახელი.

გაფრთხილება: თუ შექმნით უკვე კომპილატორში ჩაშენებული გამშვები ფაილის მსგავსი დასახელების ფაილს უმჯობესია ასეთი ფაილი არ შეინახოთ ჩაშენებული გამშვები ფაილების საქაღალდეში.

სასურველია: `#include` დირექტივა გამოძახებული იქნას `main()` ფუნქციამდე.

ჩაშენებული გამშვები ფაილები

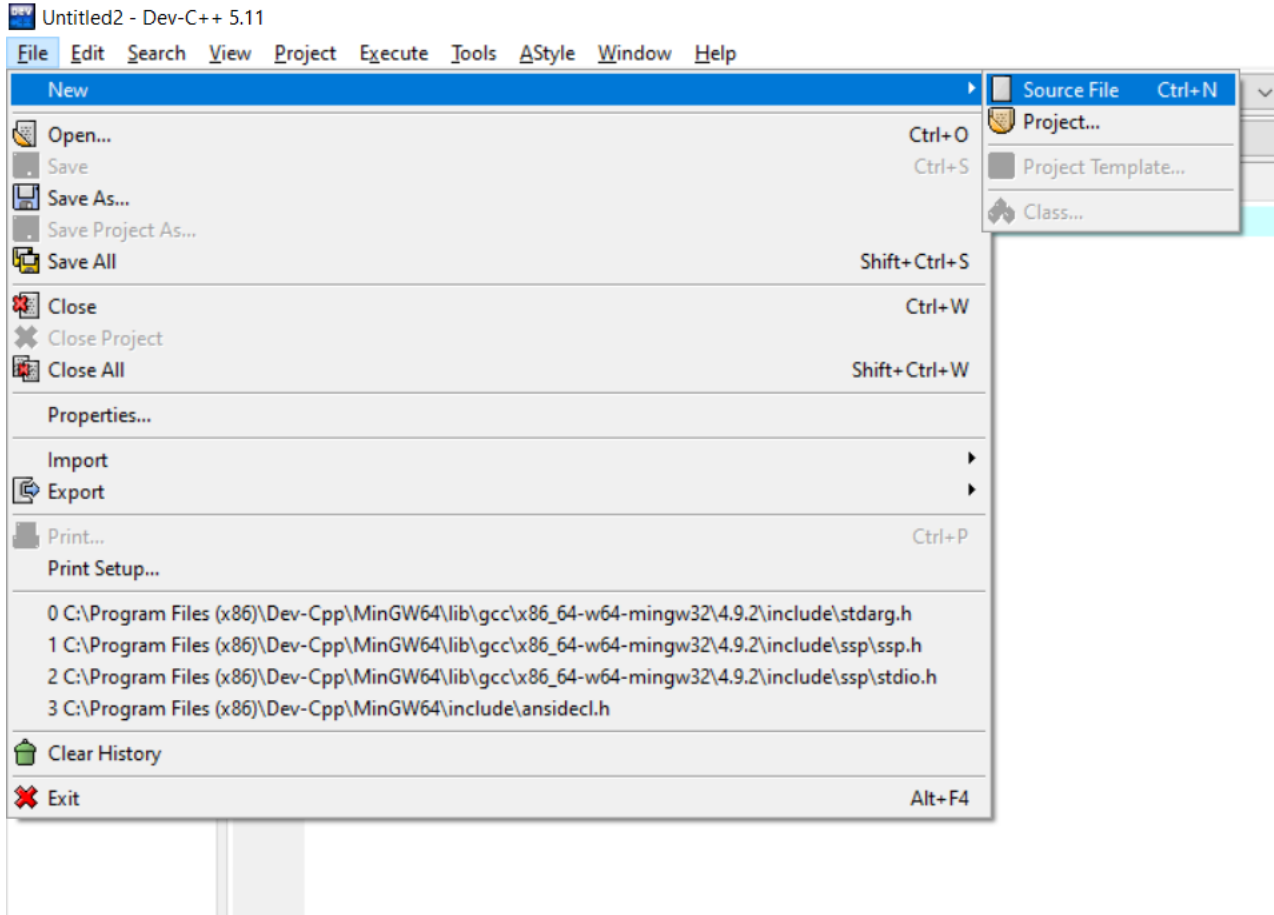
კომპილატორში ჩაშენებული გამშვები ფაილების სანახავად გახსენით:

C: Program Files -> Dev-Cpp -> MinGW64 -> include

C: Program Files -> Dev-Cpp -> MinGW64 -> lib -> gcc -> x86_64_w64_mingw32-
>4.9.2 -> include

C: Program Files -> Dev-Cpp -> MinGW64 -> lib -> gcc -> x86_64_w64_mingw32-
>4.9.2 -> include -> ssp

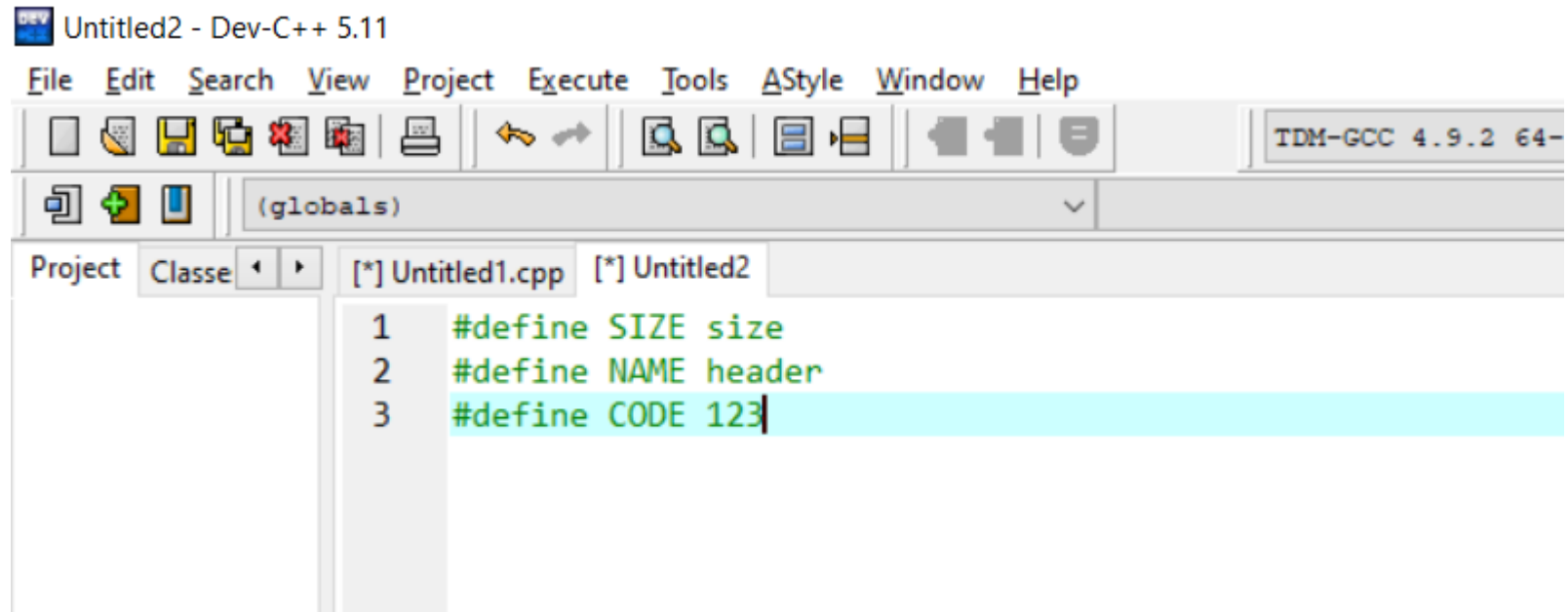
გამშვები ფაილის შექმნა



პირველ რიგში შექმენით
ახალი Source File

გამშვები ფაილის შექმნა

თუ კი გამშვები
ფაილისთვის უკვე იცით
ისეთი მონაცემები,
რომლებიც არასდროს არ
შეიცვლება შეგიძლიათ
გამოიყენოთ **#define**



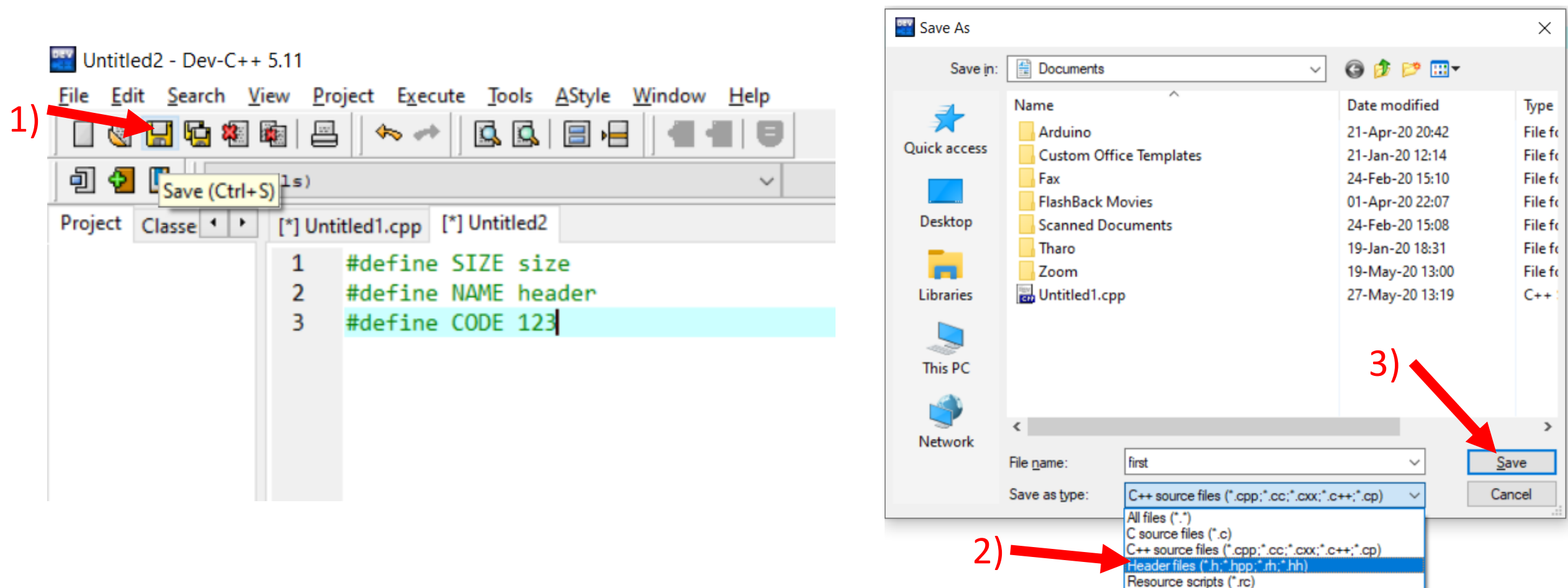
The screenshot shows the Dev-C++ 5.11 IDE interface. The title bar reads "Untitled2 - Dev-C++ 5.11". The menu bar includes File, Edit, Search, View, Project, Execute, Tools, AStyle, Window, and Help. The toolbar contains icons for file operations and execution. The status bar at the bottom right indicates "TDM-GCC 4.9.2 64-". The main editor window displays the contents of "Untitled1.cpp" with the following code:

```
1 #define SIZE size
2 #define NAME header
3 #define CODE 123
```

The third line of code is highlighted in light blue.

გამშვები ფაილის შექმნა

როდესაც დაასრულებთ გამშვები ფაილისათვის შესაბამისი ბრძანებებისა და ფუნქციების გაწერას ის უნდა შეინახოთ და ფაილს დაარქვით თქვენთვის სასურველი სახელი, რომელიც დაბოლოვდება .h გაფართოებით



ჩვენს მიერ შექმნილი გამშვები ფაილის გამოყენება

```
Untitled1.cpp  first.h
1  #include <stdio.h>
2
3  #include "first.h"
4
5  int main(){
6      printf("\n%d", SIZE );
7      printf("\n%s", NAME );
8      printf("\n%d", CODE );
9
10     return 0;
11 }
12
```




```
Untitled1.cpp  first.h
1  #define SIZE 100
2  #define NAME "salome"
3  #define CODE 1235
```

```
C:\Users\salone\Documents\Untitled1.exe

100
salome
1235
-----
Process exited after 0.03323 seconds with return value 0
Press any key to continue . . .
```

პრეპროცესორის დირექტივები

პრეპროცესორის დირექტივები შეიძლება მოიცავდეს სამი კატეგორიის ინფორმაციას:

1. მაკროსებს  #define და #undef
2. ფაილურ ჩანართს  #include
3. შედარების ოპერაციებს  #if, #ifdef, #ifndef, #elif, #else და #endif

ასევე გამაფრთხილებელ დირექტივებსაც  #error, #line, #pragma

#define

რაც შეეხება define. ის განსაზღვრავს მუდმივ ცვლადებს. ისეთი ცვლადები რომელთა მნიშვნელობა არ იცვლება პროგრამული კოდის მუშაობის დროს.

#define AGELIMIT 21

#define MYNAME "Salome Oniani"

როდესაც define-ით აღწერთ ცვლად, ცვლადის სახელი აუცილებლად უნდა იყოს Capital Letter ფორმატში

#define PI 3.14159

ცვლადის სახელი

ცვლადის
მნიშვნელობა



დავწეროთ ყვავილების რეგისტრაციის პროგრამა. სადაც შესაძლებელი იქნება მაქსიმუმ 500 ყვავილის რეგისტრაცია. ყვავილების მონაცემთა ბაზისათვის მოიფიქრეთ მინიმუმ 5 პარამეტრი. სისტემას უნდა ჰქონდეს შემდეგი მენიუ და ფუნქციები: 1. ახალი ყვავილის დამატება; 2. ყვავილების სიის გამოტანა; 3. სასურველი ყვავილის მოძებნა და 4. კონკრეტული ყვავილისათვის მონაცემების რედაქტირება.



დავწეროთ პროგრამა, რომელიც გამოიყენებს h
ფაილს სადაც გაწერილი იქნება ფუნქცია, რომელიც
დაადგენს მომხმარებლის შემოტანილი რიცხვი
მარტივია თუ არა

prime.h maincode.c prime.c

```
1 #include "prime.c"
2 int isPrime(int n);
```



prime.h prime.c maincode.c

```
1 int isPrime(int n) {
2
3     if (n<2) return 0;
4     if (n==2) return 1;
5     if ((n % 2)==0) return 0;
6
7     int i;
8     for (i=3; i<=(n/2+1); i++) {
9         if ((n % i) == 0) {
10             return 0;
11         }
12     }
13     return 1;
14 }
```

prime.h prime.c maincode.c

```
1 #include <stdio.h>
2 #include "prime.h"
3
4 int main(){
5
6     printf("please enter number ");
7     int n;
8     scanf("%d",&n);
9
10    if(isPrime(n)) printf("%d is prime number", n);
11    else printf("%d is not prime number", n);
12
13
14    return 0;
15 }
16
17
```



დავწეროთ პროგრამა, რომელიც გამოიყენებს h
ფაილს სადაც გაწერილი იქნება ნებისმიერი სიგრძის
მასივის ელემენტების ჯამის გამოსათვლელი
ფუნქცია



```
[*] prime.h  prime.c  maincode.c
1  #include "prime.c"
2  int isPrime(int n);
3  int sumArrayElements(int arr[], int size);
4
5
```

```
[*] prime.h  prime.c  maincode.c
1  int isPrime(int n) {
2
3
4      if (n<2) return 0;
5      if (n==2) return 1;
6      if ((n % 2)==0) return 0;
7
8      int i;
9      for (i=3; i< (n/2+1); i++) {
10         if ((n % i) == 0) {
11             return 0;
12         }
13     }
14     return 1;
15 }
16
17
18 int sumArrayElements(int arr[], int size){
19     int sum=0;
20     int i;
21     for (i=0; i< size; i++) {
22         sum += arr[i];
23     }
24     return sum;
25 }
```

```
[*] prime.h  prime.c  maincode.c
1  #include <stdio.h>
2  #include "prime.h"
3
4  int main(){
5
6      printf("please enter number how many elemets do you want ");
7      int n;
8      scanf("%d",&n);
9
10     int arr[n];
11     int i;
12     for (i=0; i<n; i++) {
13         printf("Enter element ");
14         scanf("%d", &arr[i]);
15     }
16
17     printf("the sum of elements is %d", sumArrayElements(arr, n));
18
19
20     return 0;
21 }
22
```



დავწეროთ პროგრამა, რომელიც გამოიყენებს h
ფაილს სადაც გაწერილი იქნება ნებისმიერი სიგრძის
მასივის ელემენტებიდან მაქსიმალური ელემენტის
პოვნის ფუნქციას



დავწეროთ პროგრამა, რომელიც გამოიყენებს h
ფაილს სადაც გაწერილი იქნება ფუნქცია, რომელიც
დაადგენს მომხმარებლის მიერ შეტანილი რიცხვი
ორის ხარისხია თუ არა



დავწეროთ პროგრამა, რომელიც გამოიყენებს h
ფაილს სადაც გაწერილი იქნება ნებისმიერ
ელემენტთან მასივში რიცხვის მოძებნის ფუნქცია.
ანუ თუ საძიებელი რიცხვი არის მასივში მაშინ
დააბრუნოს ამ რიცხვის ინდექსი, ხოლო თუ
საძიებელი რიცხვი არ აღმოჩნდება მასივში
დააბრუნოს -1 და პროგრამამ გამოიტანოს
შეტყობინება რომ ასეთი რიცხვი არ არსებობს
მასივში



დავწეროთ პროგრამა, რომელიც გამოიყენებს h
ფაილს, სადაც გაწერილი იქნება ნებისმიერ თანრიგა
რიცხვში უმაღლესი ციფრის პოვნის ფუნქცია

გმადლობთ ყურადღებისთვის!