## Задача 1:

* Създайте собствен профил в **GitHub**
* Добавете **ssh ключ** за автентикация
* Създайте организирана директория с кода, който сте написали по време на обучението. Примерна структура е:
  + Homeworks
  + Exams
  + Exercices
* Създайте **локално git repository** в новосъздадената директория
* Направете така, че **executable файлове** да не се следят от git (проучете какво е **.gitignore**)
* Направете **commit**
* Качете локалното repository в **remote** (GitHub).
* Следете кодовете, които пишете по време на обучението с помощта на git.

## Задача 2:

Създайте функция, която намира максималния елемент в масив:

**void\* findMax(void\* arr, size\_t n, size\_t elem\_s, int (\*cmp)(const void\*, const void\*));**

Функцията трябва да може да намира максималния елемент в масиви от всякакъв тип:

* **void\* arr** - адресът на първия елемент в масива
* **size\_t n** - броя на елементите в масива
* **size\_t elem\_s** - размерът на един елемент в байтове
* **int (\*cmp)(const void\*, const void\*) -** указател към функция, която приема адресите на 2 елемента от масива **arr.** Функцията сравнява елементите и:
  + връща 0 ако са еднакви
  + връща -1 ако първия аргумент е “по-голям”
  + връща 1 ако втория аргумент е “по-голям”

Като резултат функцията връща адреса на елемента, който е “най-голям”. Наредбата на елементите е определена от сравняващата функция **cmp.**

Функцията трябва да може да намира максималния елемент на масиви от всякакви типове.

### Примерно извикване

**int cmpInt(void\* a, void\* b) {**

**return \*(int\*)a > \*(int\*)b ? -1 : 1;**

**}**

**int main() {**

**int arr[] = {1,2,-1,3,5,11,12,13,2,100,22};**

**int\* max = findMax(arr, sizeof(arr)/sizeof(int), sizeof(int), cmpInt);**

**printf("%d", \*max); // should print 100**

**return 0;**

**}**

## Задача 3:

Използвайте функцията **findMax** с подходящи **comparator** функции**:**

* За да намерите най-големия елемент в масив от тип **uint64\_t**.
* За да намерите точката, която е на най-голямо (евклидово) разстояние от (0, 0, 0) в масив от тип **Point**.

**typedef struct Point {**

**double x;**

**double y;**

**double z;**

**} Point;**

* За да намерите човека с най-нисък **Body Mass Index** в масив от тип **Human.**

**typedef struct Human {**

**double weight\_kg;**

**uint16\_t height\_cm;**

**} Human;**

**Body Mass Index = weight\_kg / (height\_meter\*height\_meters)**

* За да намерите числото с най-много битове в двоичното представяне, които са 1, в масив от тип **float**.

Направете примерни извиквания в **main**.

## Задача 4

Направете функции, които обработват масив от 32 битови маски - **maskArr**. **nitems** показва броя на елементите в масива, а **bit**, показва номера на бита, който да се обработи. При успех - функциите връщат 0. Ако има проблем, функциите връщат **-1**.

**int clearBits(uint32\_t\* maskArr, size\_t nitems, size\_t bit\_index);**

* За всички елементи на масива **maskArr** се задава стойността на бита с индекс **bit\_index** да бъде 0.

**int setBits(uint32\_t\* maskArr, size\_t nitems, size\_t bit\_index);**

* За всички елементи на масива **maskArr** се задава стойността на бита с индекс **bit\_index** да бъде 1.

**int checkBits(uint32\_t\* maskArr, size\_t len, size\_t bit\_index);**

* Връща 0 ако всички елементи на **maskArr** имат стойност 0 на бита с индекс **bit\_index**
* Връща 1 ако всички елементи на **maskArr** имат стойност 1 на бита с индекс **bit\_index**
* Ако бита с индекс **bit\_index** има стойност 1 за някои елементи на **maskArr** и стойност 0 за други елементи на **maskArr** - това се счита за неуспех и функцията връща -1.

## Задача 5

Създайте структурата **MyTime**, която пази информация за :

* година
* месец
* ден
* час
* минути
* секунди

Изполвайте подходящи типове и член данни, така че размерът на структурата да е възможно най-малък. Изпечатайте размера на структурата в **main.**

Реализирайте следните функции:

* **int isLeapYear(MyTime t)** - връща 1 ако годината е високосна и 0 ако годината не е високосна
* **int totalDaysInYear(MyTime t)** - връща броя на дните в годината
* **void printMonth(MyTime t)** - печата символен низ, отговарящ на месеца
* **int secondsPassed(MyTime t)** - печата общия брой изминали секунди от 1 януари 1970 година.

Направете примерни извиквания в **main**.