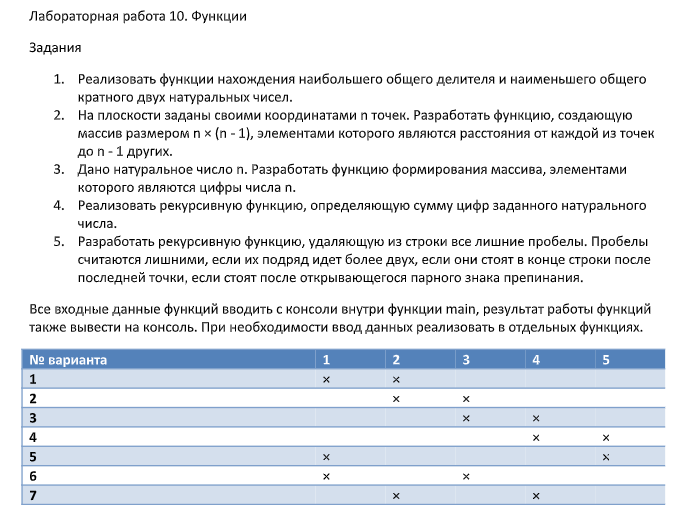
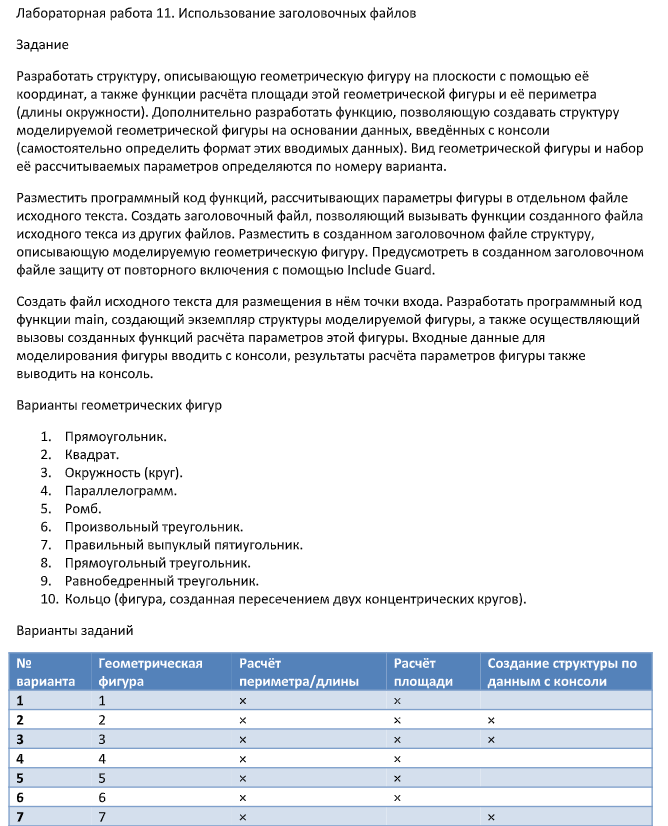
#include <stdio.h>  
#include <math.h>  
#include <string.h>  
  
int main() {  
 ///2  
 double all;  
 printf("Продолжительность разговоров: ");  
 scanf("%lf", &all);  
  
 double cost\_month;  
 printf("Абонентская плата: ");  
 scanf("%lf", &cost\_month);  
  
 double cost\_extra;  
 printf("Стоимость дополнительной минуты: ");  
 scanf("%lf", &cost\_extra);  
  
 while (cost\_extra <= (cost\_month / 499.0)) { //Если доп.плата меньше абонентской  
 printf("Стоимость дополнительной минуты должна быть больше минуты в тарифе. Попробуйте еще раз: "); //То вводим новую, пока не введется нормальная доп плата  
 scanf("%lf", &cost\_extra);  
 }  
  
 printf("Стоимость обслуживания: %0.2lfRUB\n\n", cost\_month + (fmax(all - 499.0, 0.0)\* cost\_extra)); //Абон плата + штрафные минуты (если есть) \* их цену  
  
  
 ///4  
 int maxn;  
 printf("Введите число, до которого нужно ввести простые ");  
 scanf("%d", &maxn);  
 int arr[1001]; //Делаем массив для решета Эратосфена  
 memset(arr, 0, 1001);  
 for (int i = 2; i <= maxn; i++) {  
 if (!arr[i]) { //Если наше число простое  
 printf("%d ", i); //То выводим его  
 for (int j = i \* i; j <= maxn; j += i) {  
 arr[j] = 1;  
 }  
 }  
 }  
 return 0;  
}

#include <stdio.h>  
#include <math.h>  
  
struct Point {  
 double x, y;  
};  
  
struct Point arr[101];  
double dist[10001];  
  
double distance(struct Point x, struct Point y) {  
 return sqrt((x.x - y.x) \* (x.x - y.x) + (x.y - y.y) \* (x.y – y.y)); //Корень суммы

// квадратов разностей иксов и игриков  
}  
  
int sum\_of\_digits(int x) {  
 if (!x) return 0; //Если наше число = 0, то просто возратим 0  
 return (x % 10) + sum\_of\_digits(x / 10); //Иначе возвращаем последнюю цифру и вызываем //эту функцию для числа без последней цифрой  
}  
  
int main() {  
 ///2  
 int n;  
 scanf("%d", &n);  
 for (int i = 0; i < n; i++) {  
 scanf("%lf%lf", &arr[i].x, &arr[i].y); //Читаем точку  
 }  
 int it = 0;  
 for (int i = 0; i < n; i++) {  
 for (int j = 0; j < n; j++) {  
 if (i == j) continue;  
 dist[it++] = distance(arr[i], arr[j]); //Высчитываем расстояние  
 }  
 }  
 for (int i = 0; i < n \* (n - 1); i++) {  
 printf("%lf ", dist[i]);  
 }  
 printf("\n\n");  
  
 ///4  
 int x;  
 scanf("%d", &x);  
 printf("%d", sum\_of\_digits(x)); //Вызываем функцию для введенного числа  
}



Файл Pentagon.h

//  
// Created by vadim on 4.11.19.  
//  
  
#ifndef PENTAGON\_H  
#define PENTAGON\_H

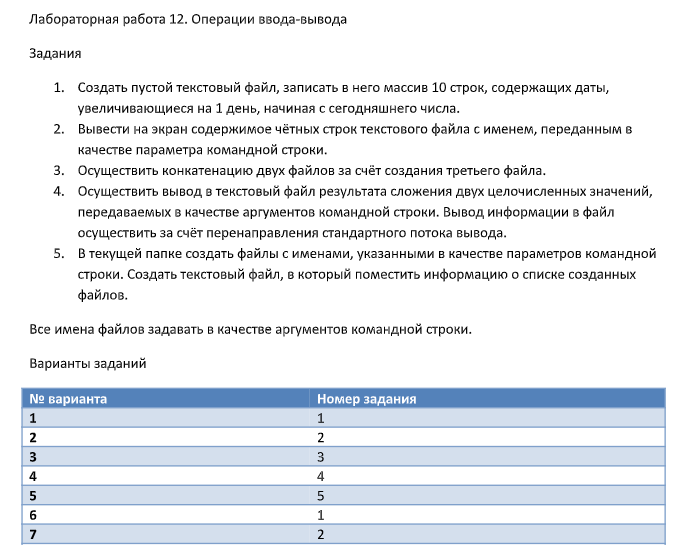
struct Pentagon {  
 double R;  
};  
  
double perimeter(struct Pentagon x);  
void init(struct Pentagon\* x, double R);  
  
  
#endif //PENTAGON\_H

Файл Pentagon.c

//  
// Created by vadim on 4.11.19.  
//  
  
#include "Pentagon.h"  
#include <math.h>  
  
double perimeter(struct Pentagon x) {  
 double t = x.R \* sqrt(5 - sqrt(5)) / 2;  
 return t \* 5;  
}  
  
void init(struct Pentagon\* x, double R) {  
 x->R = R;  
 return;  
}

Файл main.c

#include <stdio.h>  
#include "Pentagon.h"  
  
int main() {  
 struct Pentagon x;  
 double R;  
 scanf("%lf", &R);  
 init(&x, R);  
 printf("Perimeter = %lf", perimeter(x));  
}

#include <stdio.h>  
#include <string.h>  
  
int main(int argc, char\*\* argv) {  
 if (argc < 2) { // Первый элемент в аргументах – путь самого exe, далее уже идет нужная

//нам строка с именем файла, откуда мы будем считывать  
 printf("Something went wrong :(\n"); // Нам ничего не передали  
 return 0;  
 }  
 char\* filename = argv[1];  
 FILE\* fp = fopen(filename, "r");  
 char\* line = NULL;  
 size\_t len = 0;  
 ssize\_t read;  
  
 if (fp == NULL) { // Такого файла не существует  
 printf("I think such file does not exist :(\n");  
 return 0;  
 }  
  
 int it = 1;  
 while ((read = getline(&line, &len, fp)) != -1) { //Читаем новую строку из файла, пока

//они есть  
 if (!(it & 1)) printf("line = %s", line); //Если это четная строка, то выводим  
 it++;  
 }  
}