МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ ИМ. Р.Е. АЛЕКСЕЕВА»

(НГТУ)

Институт радиоэлектроники и информационных технологий

Кафедра «Информационные радиосистемы»

**Контрольная работа по дисциплине  
«Информационные технологии»**

Направление подготовки: \_\_\_\_11.03.01 Радиотехника\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*код и наименование направления подготовки*

Выполнил:

Студент гр. 24-Рз\_\_\_\_\_\_\_\_\_\_\_\_Иванов И.И.

*(группа) (подпись)*

Проверил:

доцент кафедры ИРС \_\_\_\_\_\_\_\_\_\_ Балашова Д.М.

*(подпись)*

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_

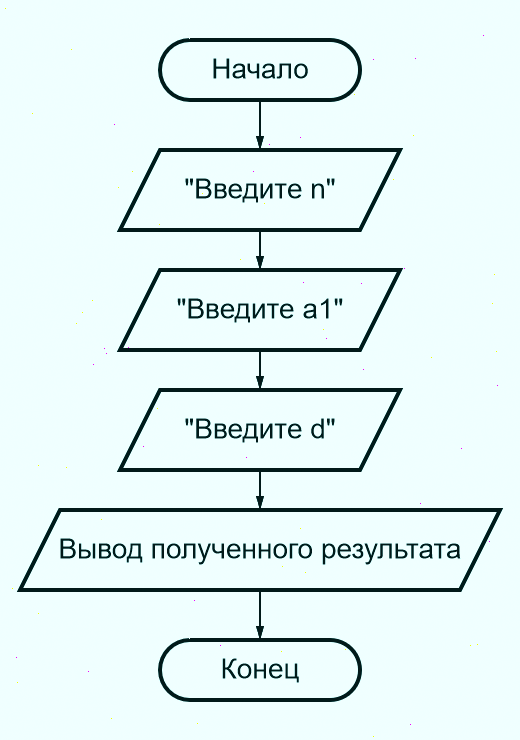
Дата: «\_\_\_» \_\_\_\_\_\_\_\_\_\_2024 г.

Нижний Новгород, 2024

**Задание 1**

**Вариант 10**

Вычислить сумму первых *n*-членов арифметической прогрессии по формуле , где *n* - количество членов прогрессии, *an —* *n*-й член прогрессии, *a1* - первый член прогрессии. Параметры должны вводиться с клавиатуры.

**Блок-схема алгоритма**

**Листинг программного кода**

#include <stdio.h>

int main(){

    double n, d, a1, summ;

    printf("Input n: ");

    scanf("%lf", &n);

    printf("Input a1: ");

    scanf("%lf", &a1);

    printf("Input d (sequence diviation): ");

    scanf("%lf", &d);

    summ = n/2\*(2\*a1+(n-1)\*d);

    printf("Summ is: %.f\n", summ);

    return 0;

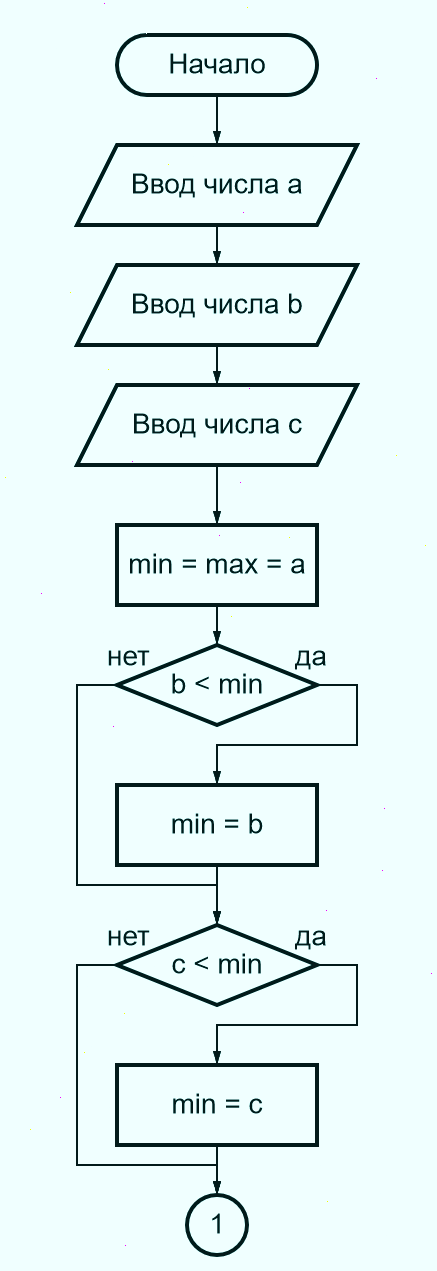
}

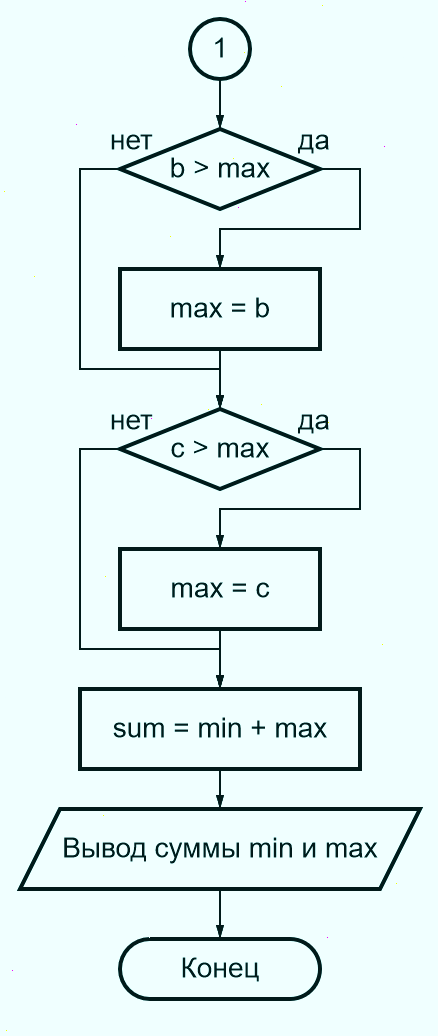
**Задание 2. Часть 1**

**Вариант 8**

Даны три числа. Найти сумму большего и меньшего чисел из этих трех.

**Блок-схема алгоритма**

****

****

**Листинг программного кода**

#include <stdio.h>

int main(){

    double a, b, c;

    double min, max, summ;

    printf("Input a:\n");

    scanf("%lf", &a);

    printf("Input b:\n");

    scanf("%lf", &b);

    printf("Input c:\n");

    scanf("%lf", &c);

    min = max = a;

    if(b < min){

        min = b;

    }

    if(c < min){

        min = c;

    }

    if(b > max){

        max = b;

    }

    if(c > max){

        max = c;

    }

    summ = min + max;

    printf("SUMM IS: %.f\n", summ);

    return 0;

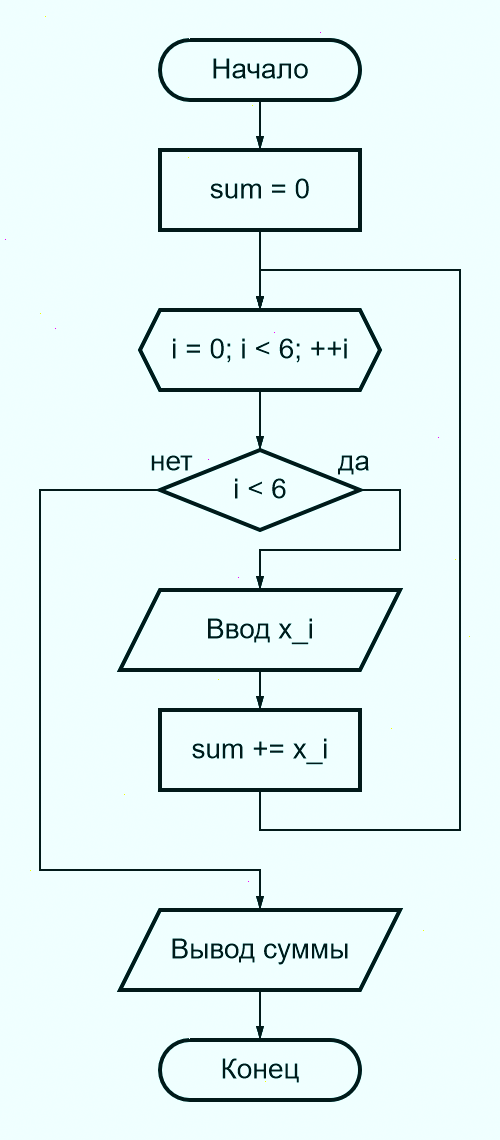
}

**Задание 2. Часть 2**

**Вариант 5**

Вычислить сумму *S* членов последовательности действительных чисел *xi*, где i = 0, 1,…, 5.  ***.***

**Блок-схема алгоритма**

****

**Листинг программного кода**

#include <stdio.h>

int main(){

    double x[6];

    double sum = 0;

    for (int i = 0; i < 6; ++i){

        printf("Enter value of an x[%d]:\n", i);

        scanf("%lf", &x[i]);

        sum += x[i];

    }

    printf("Summ is: %.2lf\n", sum);

    return 0;

}

**Задание 3**

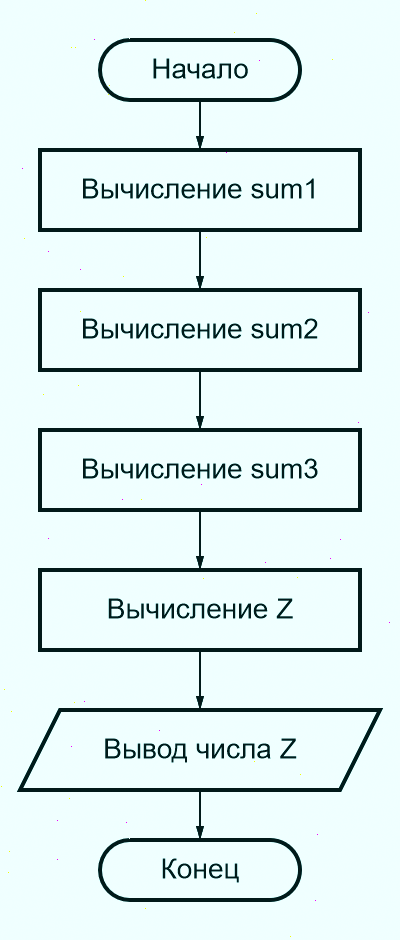
**Вариант 4**

Формула: ***;***

Функция:

где b=1; A = 0; B = 9; C = 1.

Расчет сумм в формуле и расчет Yk должны быть оформлены в виде отдельных функций.

**Блок-схема алгоритма**

**Листинг программного кода**

#include <stdio.h>

#include <math.h>

double calculate\_YK(int K, double b, double A, double C, double B) {

    double numerator = log(10.0\*(A\*K+C));

    double denominator = sqrt(K+A+B);

    return b \* (numerator / denominator);

}

double calculate\_sum(int start, int end, double b, double A, double C, double B) {

    double sum = 0.0;

    for (int K = start; K <= end; K++) {

        sum += calculate\_YK(K, b, A, C, B);

    }

    return sum;

}

int main() {

    double b = 1, A = 0, B = 9, C = 1;

    double sum1 = calculate\_sum(3, 10, b, A, C, B);

    double sum2 = calculate\_sum(6, 20, b, A, C, B);

    double sum3 = calculate\_sum(11, 30, b, A, C, B);

    double Z = sin(sum1)+B\*cos(sum2)+(C/sum3);

    printf("Z = %lf\n", Z);

    return 0;

}

**Задание 4 - 1**

**Вариант 2**

Переставить элементы введенной текстовой строки в обратном порядке. Длина строки не более 80 символов. Строка должна вводиться с клавиатуры.

**Листинг программного кода**

**Задание 4 - 2**

**Вариант 2**

Переставить элементы введенной текстовой строки в обратном порядке. Длина строки не более 80 символов. Строка должна вводиться с клавиатуры.

**Листинг программного кода**

4-functions.h

#ifndef FUNCTIONS\_H

#define FUNCTIONS\_H

void get\_input(char \*str, int length);

void process(char \*str);

void do\_output(const char \*str);

#endif

4- main.c

#include <stdio.h>

#include "4-functions.h"

int main() {

    char str[81];

    get\_input(str, sizeof(str));

    process(str);

    do\_output(str);

    return 0;

}

4 – input.c

#include <stdio.h>

#include "4-functions.h"

void get\_input(char \*str, int length) {

    printf("Enter a string (up to %d characters): ", length - 1);

    fgets(str, length, stdin);

    str[strcspn(str, "\n")] = 0;

}

4 – process.c

#include <string.h>

#include "4-functions.h"

void process(char \*str) {

    int length = strlen(str);

    for (int i = 0; i < length / 2; i++) {

        char temp = str[i];

        str[i] = str[length - 1 - i];

        str[length - 1 - i] = temp;

    }

}

4 – output.c

#include <stdio.h>

#include "4-functions.h"

void do\_output(const char \*str) {

    printf("Reversed string: %s\n", str);

}

**Задание 5 - 1**

**Вариант 6**

Таблица содержит информацию о кадрах некоторой организации. Информация о каждом сотруднике включает его ФИО, должность, табельный номер и номер отдела, в котором он работает. Зная количество отделов в организации, подсчитать, сколько сотрудников работает в каждом из них.

**Листинг программного кода**

5–array–header.h

#ifndef HEADER\_H

#define HEADER\_H

#include <stdio.h>

#define NAME\_MAX\_SIZE 50

#define HR\_COUNT 2

#define DEPT\_COUNT 4

typedef struct Personnel{

    char name[NAME\_MAX\_SIZE];

    char position[NAME\_MAX\_SIZE];

    int id;

    int department;

}Personnel;

Personnel get\_input();

void do\_process(Personnel employees[], int known\_departments[], int departments\_count[]);

void do\_output(int known\_departments[], int departments\_count[]);

#endif //HEADER\_H

5–array–main.c

#include "5-array-header.h"

int main(){

    Personnel employees[HR\_COUNT];

    int known\_departments[DEPT\_COUNT] = {101, 102, 103, 104};

    int department\_counts[DEPT\_COUNT] = {0};

    for (int i = 0; i < HR\_COUNT; ++i) {

        printf("\n--- Enter details for employee %d ---\n", i + 1);

        employees[i] = get\_input();

    }

    do\_process(employees, known\_departments, department\_counts);

    do\_output(known\_departments, department\_counts);

    return 0;

}

5–array–input.c

#include "5-array-header.h"

    // MEMO

    // char name[NAME\_MAX\_SIZE];

    // char position[NAME\_MAX\_SIZE];

    // int id;

    // int department;

Personnel get\_input() {

    Personnel person;

    printf("\nInput a name: ");

    scanf("%49s", person.name);

    printf("Input a position: ");

    scanf("%49s", person.position);

    printf("Input an ID: ");

    scanf("%d", &person.id);

    printf("Input a department: ");

    scanf("%d", &person.department);

    return person;

}

5–array–process.c

#include "5-array-header.h"

void do\_process(Personnel employees[], int known\_departments[], int department\_counts[]) {

    for (int i = 0; i < HR\_COUNT; ++i) {

        for (int j = 0; j < DEPT\_COUNT; ++j) {

            if (employees[i].department == known\_departments[j]) {

                department\_counts[j]++;

                break;

            }

        }

    }

}

5–array–output.c

#include "5-array-header.h"

void do\_output(int known\_departments[], int department\_counts[]){

    printf("\nNumber of employees per department:\n");

    for (int i = 0; i < DEPT\_COUNT; ++i) {

        printf("Department %d has %d employee(s)\n", known\_departments[i], department\_counts[i]);

    }

}

**Задание 5 - 2**

**Вариант 6**

Таблица содержит информацию о кадрах некоторой организации. Информация о каждом сотруднике включает его ФИО, должность, табельный номер и номер отдела, в котором он работает. Зная количество отделов в организации, подсчитать, сколько сотрудников работает в каждом из них.

**Листинг программного кода**

5–pointer–header.h

#ifndef HEADER\_H

#define HEADER\_H

#include <stdio.h>

#define NAME\_MAX\_SIZE 50

#define HR\_COUNT 5

#define DEPT\_COUNT 4

typedef struct Personnel{

    char name[NAME\_MAX\_SIZE];

    char position[NAME\_MAX\_SIZE];

    int id;

    int department;

}Personnel;

typedef struct Company{

    Personnel personnel[HR\_COUNT];

    int known\_departments[DEPT\_COUNT];

    int department\_counts[DEPT\_COUNT];

}Company;

int do\_run(Company\* company);

void do\_input(Company\* company);

void do\_process(Company\* company);

void do\_output(Company\* company);

#endif //HEADER\_H

5–pointer–main.c

#include "5-pointer-header.h"

int main(){

    Company company;

    int ret = do\_run(&company);

    return ret;

}

5-pointer-run.c

#include "5-pointer-header.h"

int do\_run(Company\* company){

    do\_input(company);

    do\_process(company);

    do\_output(company);

    return 0;

}

5-pointer-input.c

#include "5-pointer-header.h"

//DEBUG STUFF

// #include <stdlib.h>

// #include <time.h>

void do\_input(Company\* company){

    printf("Welcome to our comanys' HR indexing system!\n");

    //artificially fill departments and reset it's counters to zeroes

    printf("\n--- FYI, known departments are:");

    for (int i = 0; i < DEPT\_COUNT; i++) {

        company->known\_departments[i] = 101 + i;

        printf(" %d", company->known\_departments[i]);

        company->department\_counts[i] = 0;

    }

    // GIVE THE RAND SOME RANDOM SEED, ALSO GETTING BOUNDARIES FOR RAND()

    // srand(time(NULL));

    // int upper\_bound = company->known\_departments[DEPT\_COUNT-1];

    // int lower\_bound = company->known\_departments[DEPT\_COUNT-DEPT\_COUNT];

    // printf("\n\n%d - ub, %d - lb\n\n", upper\_bound, lower\_bound);

    //fill the employees

    for (int i = 0; i < HR\_COUNT; ++i){

        printf("\n--- Enter details for employee %d of %d ---\n", i + 1, HR\_COUNT);

        printf("\nInput a name: ");

        scanf("%49s", company->personnel[i].name);

        printf("Input a position: ");

        scanf("%49s", company->personnel[i].position);

        printf("Input an ID: ");

        scanf("%d", &company->personnel[i].id);

        printf("Input a department: ");

        scanf("%d", &company->personnel[i].department);

        // IS USED TO BATCH PSEUDO\_RANDOMLY FILL DEPARTMENTS

        // company->personnel[i].department = rand() % (upper\_bound-lower\_bound+1) + lower\_bound;

        // printf("GOD's FAVOUR IS UPON: %d", company->personnel[i].department);

    }

}

5-pointer-process.c

#include "5-pointer-header.h"

void do\_process(Company\* company){

    //count employees per department

    for (int i = 0; i < HR\_COUNT; ++i) {

        for (int j = 0; j < DEPT\_COUNT; ++j) {

            if (company->personnel[i].department == company->known\_departments[j]) {

                company->department\_counts[j]++;

                break;

            }

        }

    }

}

5-pointer-output.c

#include "5-pointer-header.h"

void do\_output(Company\* company){

    //output the results

    printf("\nNumber of employees per department:\n");

    for (int i = 0; i < DEPT\_COUNT; ++i) {

        printf("Department %d has %d employee(s)\n", company->known\_departments[i], company->department\_counts[i]);

    }

    int check\_summ\_dept = 0;

    for(int i = 0; i < DEPT\_COUNT; ++i){

        check\_summ\_dept += company->department\_counts[i];

    }

    if (check\_summ\_dept == HR\_COUNT){

        printf("SUCCESS!");

    }

    else{

        printf("FAILURE");

    }

}