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.

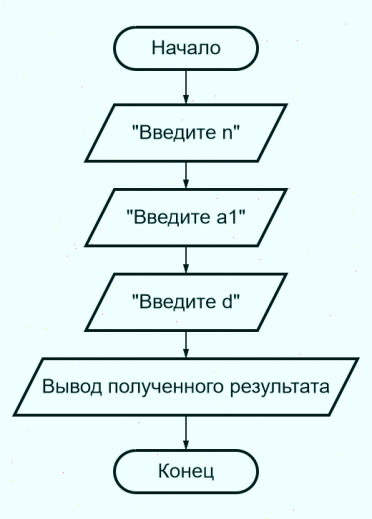
Нижний Новгород, 2024

ЗАДАНИЕ 1

ВАРИАНТ 1

Вычислить сумму первых *n*-членов арифметической прогрессии по формуле $S=\frac{n}{2}(2a_1+(n-1)d)$, где *n* - количество членов прогрессии, *d* — разность прогрессии, a_1 — первый член прогрессии. Параметры должны вводиться с клавиатуры.

Блок-схема алгоритма



Листинг программного кода

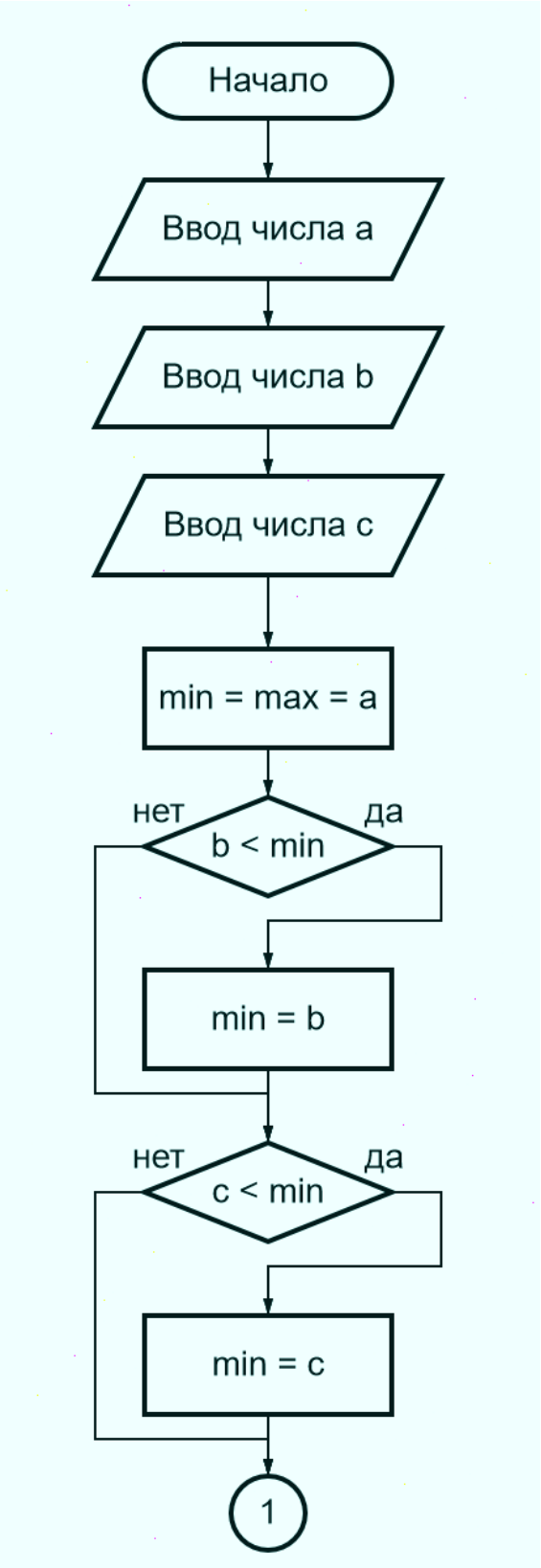
```
#include <stdio.h>

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;
}
```

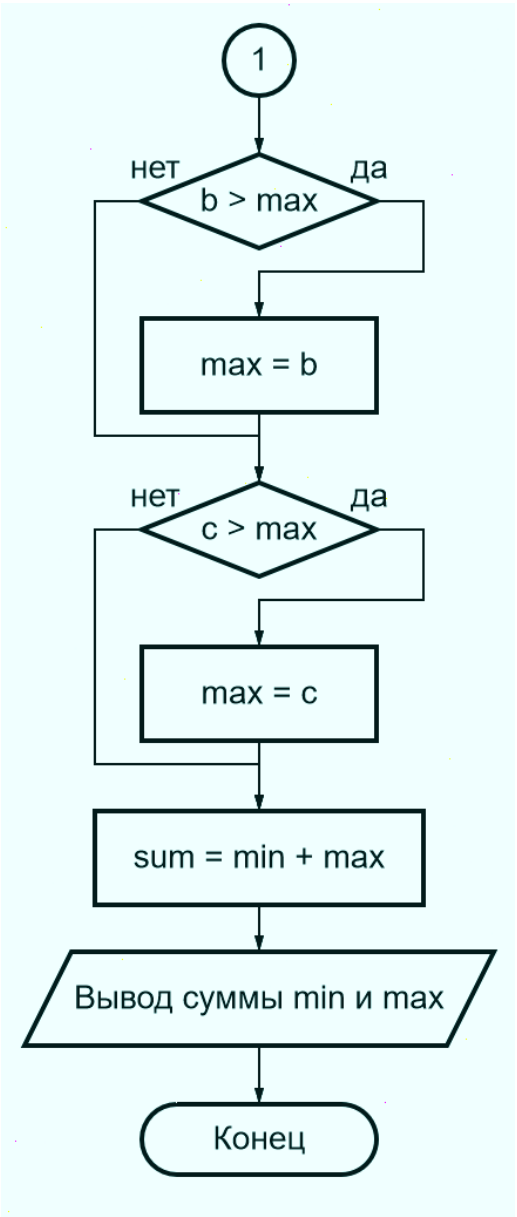
ЗАДАНИЕ 2. ЧАСТЬ 1

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);
```

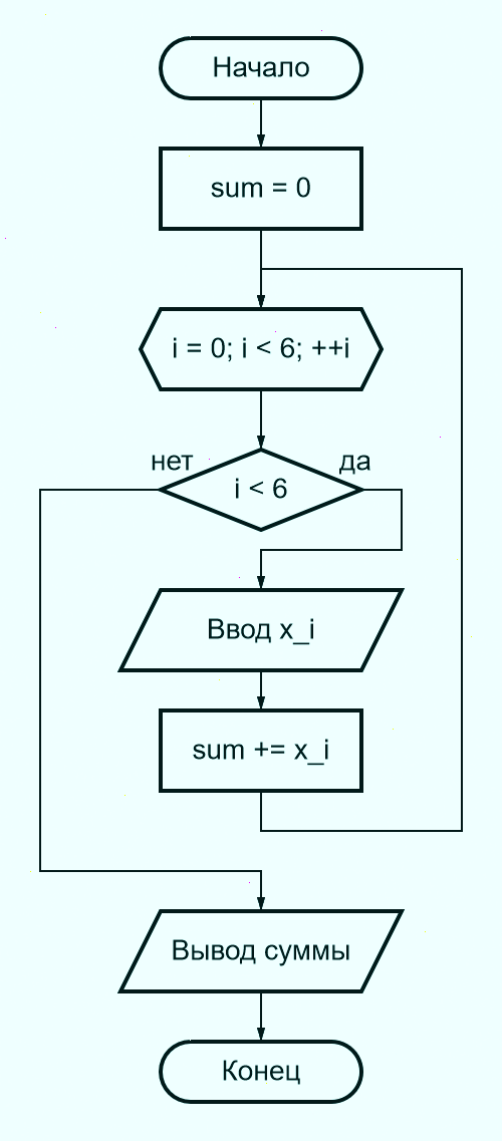
```
    min = max = a;

    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 x_i, где $i = 0, 1, \ldots, 6$ $S = \sum_{i=0}^{5} x_i$.

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

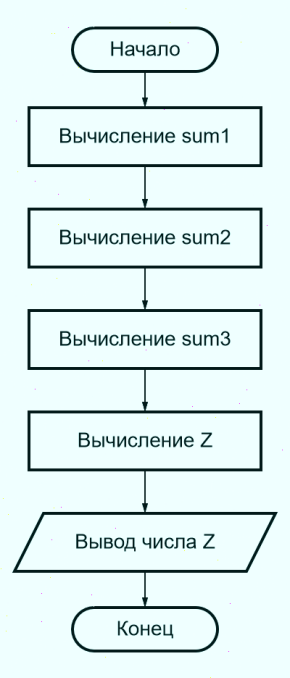
Формула: $$Z=\sin\left(\sum_{K=3}^{10} Y_K\right)+B\cdot\cos\left(\sum_{K=6}^{20} Y_K\right)+\frac{C}{\sum_{K=11}^{30} Y_K};$$

Функция: $Y_k=b*(\frac{(\ln(10*(A*K+C)))}{\sqrt{(K+A+B)}})$

где b=1; A = 0; B = 9; C = 1.

Расчет сумм в формуле и расчет Y_k 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

4-array-functions.h

```
#IFNDEF FUNCTIONS_H
#define FUNCTIONS_H

#include <string.h>
#include <stdio.h>

void get_input(char str[81]);
void do_process(char str[81]);
void do_output(char str[81]);

#endif
```

4-array-main.c

```
#include "4-array-functions.h"

int main() {

  char str[81];

  get_input(str);
  do_process(str);
  do_output(str);

  return 0;

}
```

4-array-input.c

```
#include "4-array-functions.h"

void get_input(char str[81]) {

  printf("Enter a string (up to 80 characters): ");
  fgets(str, 81, stdin);

  str[strcspn(str, "\n")] = '\0';

}
```

4-array-process.c

```
#include "4-array-functions.h"

void do_process(char str[81]) {
  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-array-output.c

```
#include "4-array-functions.h"

void do_output(char str[81]) {

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

}
```

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-pointer-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-pointer-main.c

```
#include <stdio.h>
#include "4-pointer-functions.h"

int main() {

    char str[81];

    get_input(str, sizeof(str));
    process(str);
    do_output(str);

    return 0;
}
```

4–pointer-input.c

```
#include <stdio.h>
#include "4-pointer-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–pointer-process.c

```
#include <string.h>
#include "4-pointer-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–pointer-output.c

```
#include <stdio.h>
#include "4-pointer-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

```c
#include "5-pointer-header.h"

int main(){

  Company company;
  int ret = do_run(&company);
  return ret;

}
```

5-pointer-run.c

```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

```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");
  }
}
```