Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«КУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Кафедра программного обеспечения и администрирования информационных систем

Направление подготовки математическое обеспечение и администрирование информационных систем

Форма обучения очная

**Отчет**

**по лабораторной работе №1**

«Программирование на языке C++ с использованием рекурсивных функций и

функций со сложными параметрами»

дисциплина «Основы алгоритмизации и программирования»

Выполнил:

студент группы 113 Гришанов Е.А.

Проверил:

старший преподаватель кафедры ПОиАИС Ураева Е.Е.

Курск, 2024

***Цель работы:*** Изучить особенности написания программ на языке С++ с использованием рекурсивных функций и функций со сложными параметрами.

***Задание***

*Задача 1.* Описать бестиповую функцию Time24ToTime12 (H, M, S, am), которая преобразует время, заданное в часах H, минутах M и секундах S (H, M и S — целые положительные) из 24-часового формата в 12-часовой, при этом для времени в 12-часовом формате устанавливается признак принадлежности момента времени первой половине суток am (am - логическое). Используя эту функцию, привести к новому формату два заданных временных отрезка: (T1, M1, S1) и (T2, M2, S2).

*Задача 2.* Даны два массива целых чисел размеров n и m и целые числа k и t. Если в первом массиве все элементы не превосходят k, а во втором массиве все элементы не превосходят t, то в первом массиве заменить все элементы равные k значением t, в противном случае во втором массиве заменить все элементы, равные t значением k.

*Задача 3.* Используя рекурсивную функцию, для заданного числа n определить значение выраженияx – действительное число, i = 0, 1, … n, j = 1, 2, …n – 1.

***Разработка алгоритма***

*Задача 1*

Основной алгоритм решения задачи реализован с использованием следующей функции:

*void Time24ToTime12(int\* H, int\* M, int\* S, bool\* am);*

Функция Time24ToTime12 преобразует время, заданное в часах H, минутах M и секундах S (H, M и S — целые положительные) из 24-часового формата в 12-часовой.

Входные аргументы:

*- H, M, S –* целые числа;

- аm – булево значение.

Выходные аргументы:

*- H, M, S –* целые числа.

Входные данные алгоритма решения задачи:

* *T1, M1, S1, T2, M2, S2* – целые числа.

Выходные данные алгоритма решения задачи:

* *T1, M1, S1, T2, M2, S2* – целые числа.

Алгоритм решения задачи представлен на рисунках 1, 2.

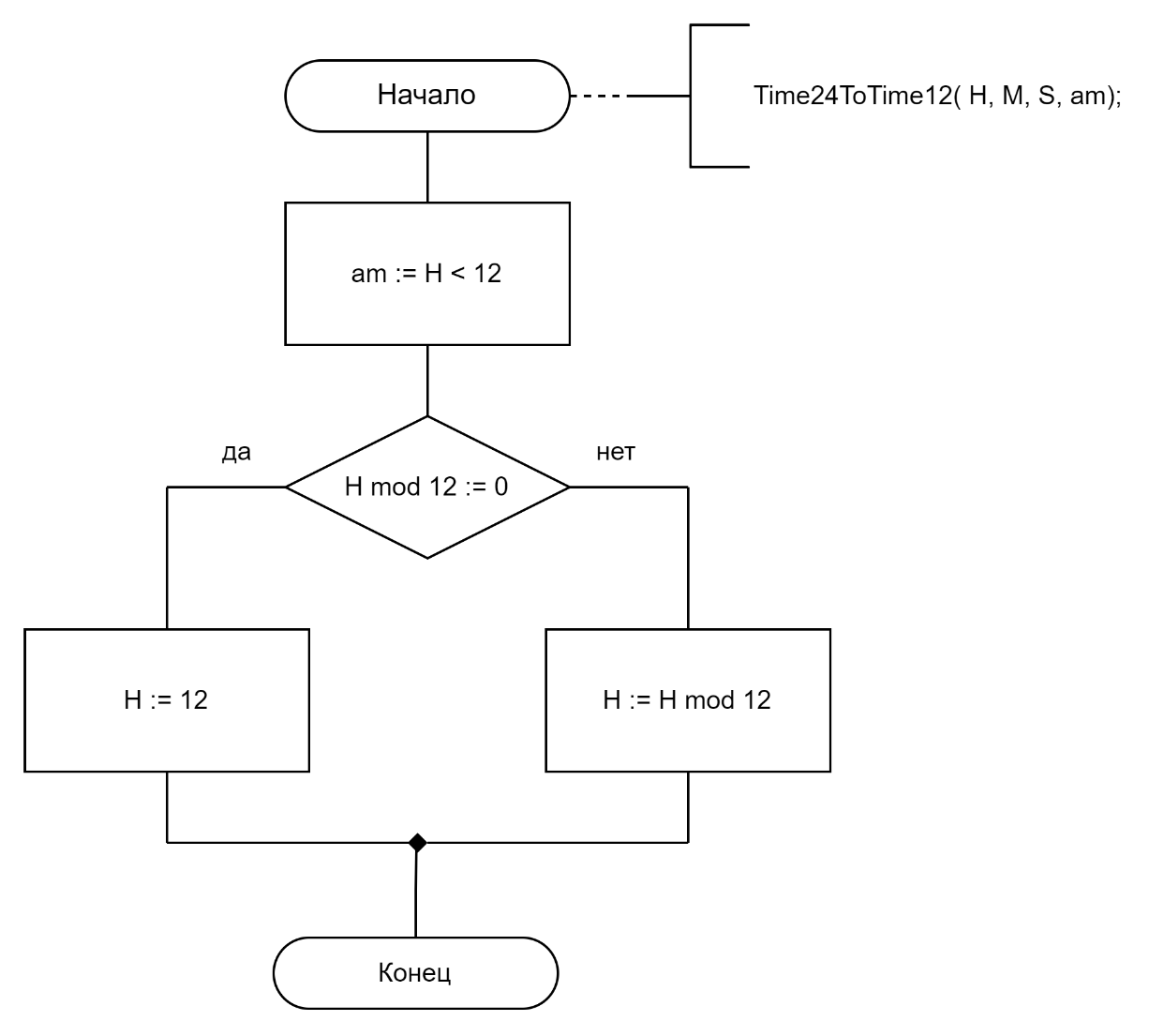


Рисунок 1 – Алгоритм функции *Time24ToTime12*

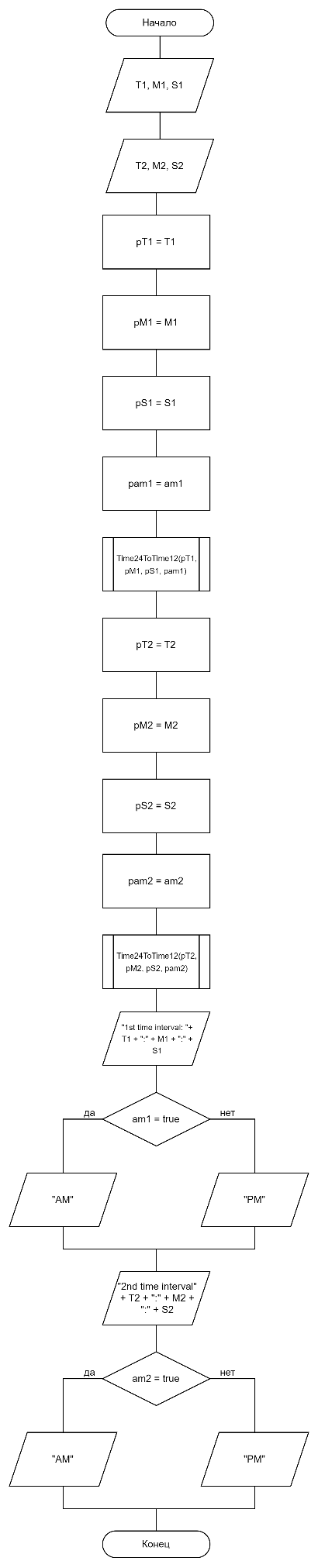


Рисунок 2 – Алгоритм решения задачи 1

*Задача 2*

Основной алгоритм решения задачи реализован с использованием следующих функций:

*bool check(int\* mas, int k);*

*void repl(int\* mas, int k, int t).*

Функция check проверяет условия, в соотвествии с которыми мы будем изменять содержимое массивов в функции repl.

Функция repl преобразует числа в массиве таким образом что, если в первом массиве все элементы не превосходят k, а во втором массиве все элементы не превосходят t, то в первом массиве заменит все элементы равные k значением t, в противном случае во втором массиве заменит все элементы, равные t значением k.

Входные аргументы:

*- mas1, mas2 –* одномерные динамические массивы;

- k, t, n, m – целые числа.

Выходные аргументы:

*- mas1, mas2 –* одномерные динамические массивы.

Входные данные алгоритма решения задачи:

* *Arr1, arr2 -* одномерные динамические массивы;
* k, t, n, m – целые числа.

Выходные данные алгоритма решения задачи:

* *Arr1, arr2 -* одномерные динамические массивы.

Алгоритм решения задачи представлен на рисунках 3, 4.

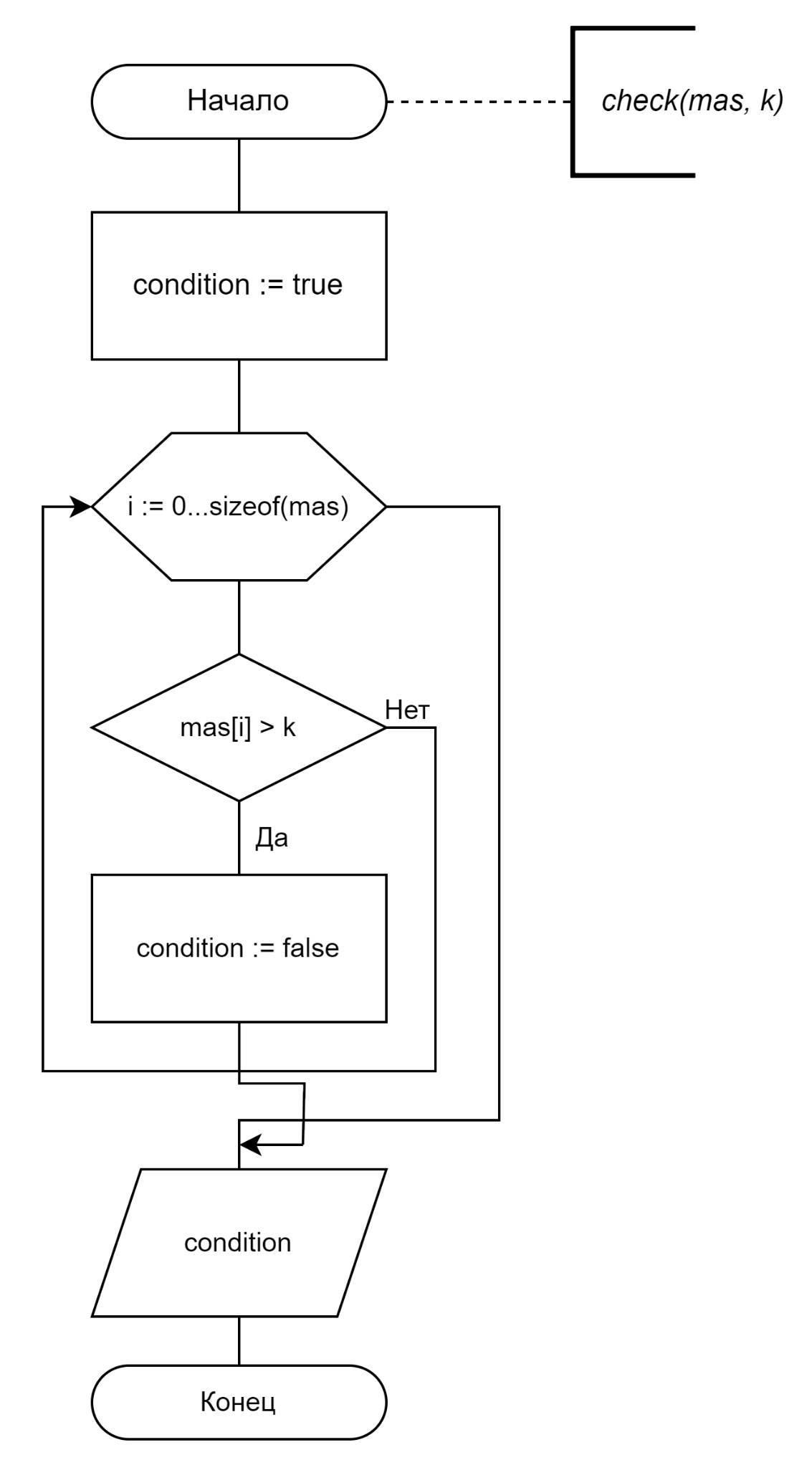


Рисунок 3 – Алгоритм функции check

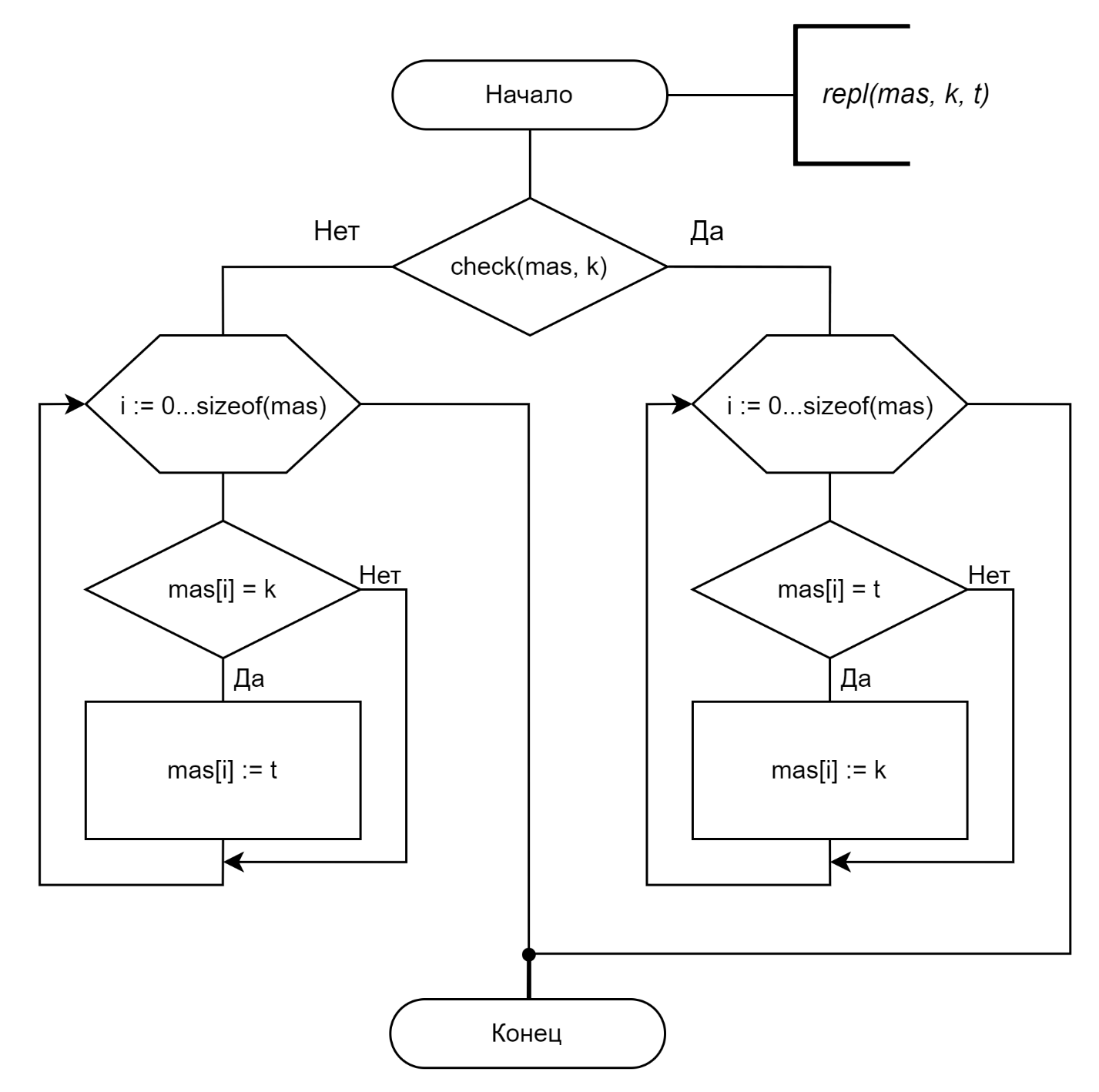


Рисунок 4 – Алгоритм функции repl

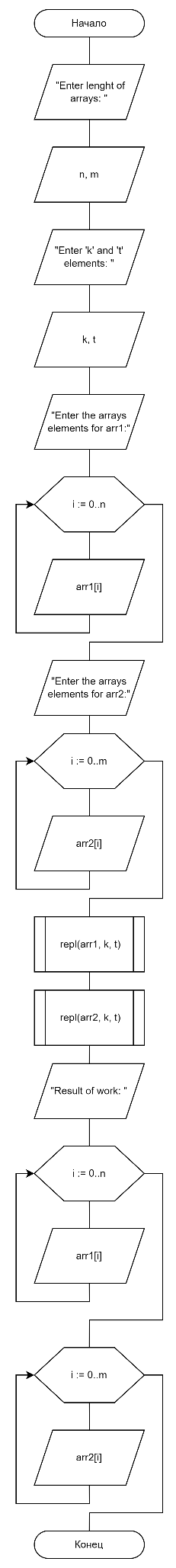


Рисунок 5 – Алгоритм решения задачи 2

*Задача 3*

Основной алгоритм решения задачи реализован косвенной рекурсией с использованием следующих функций:

*double f(double x, int i, int j);*

*double calculateSum(double x, int n, int i, int j);*

*double calculateProd(double x, int n, int i).*

Функция f возвращает настоящее значение данной математической функции .

Функция calculateSum возвращает значение , используя косвенную рекурсию функции f.

Функция calculateProd возвращает значение

Входные аргументы:

- x, n, i, j – целые числа.

Выходные аргументы:

*-* calculateSum(x, n, i, 1) + calculateProd(x, n, i + 1) – целое число.

Входные данные алгоритма решения задачи:

* x – действительное число;
* n – целое число.

Выходные данные алгоритма решения задачи:

* *result* – действительное число.

Алгоритм решения задачи представлен на рисунках 6, 7, 8, 9.

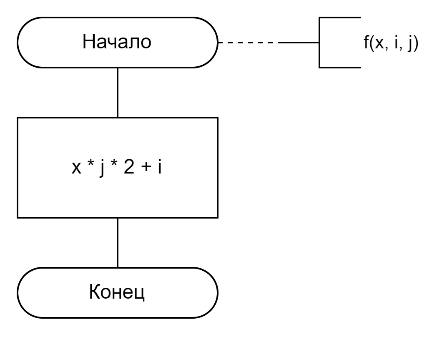


Рисунок 6 – Алгоритм функции f

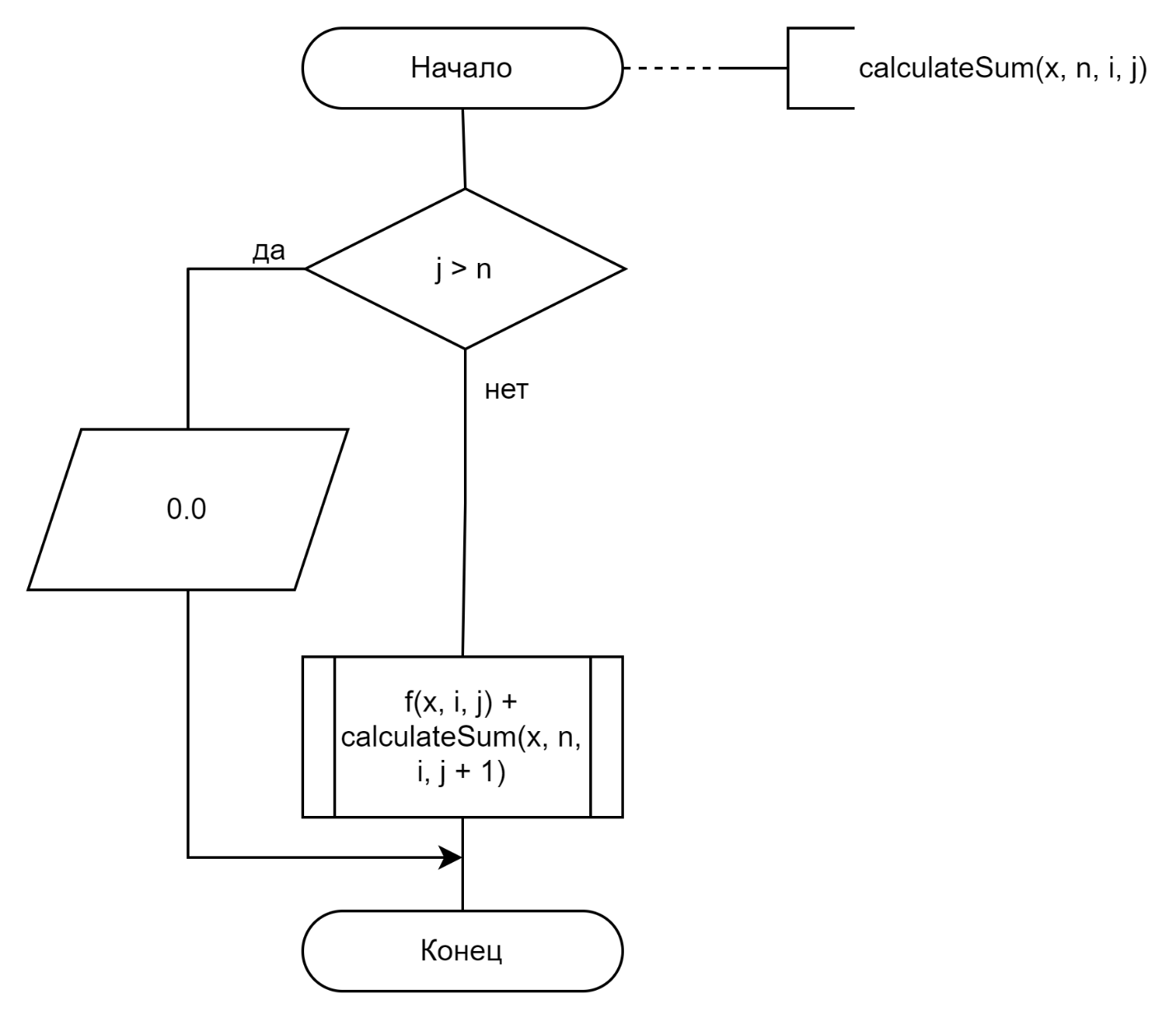


Рисунок 7 – Алгоритм функции calculateSum

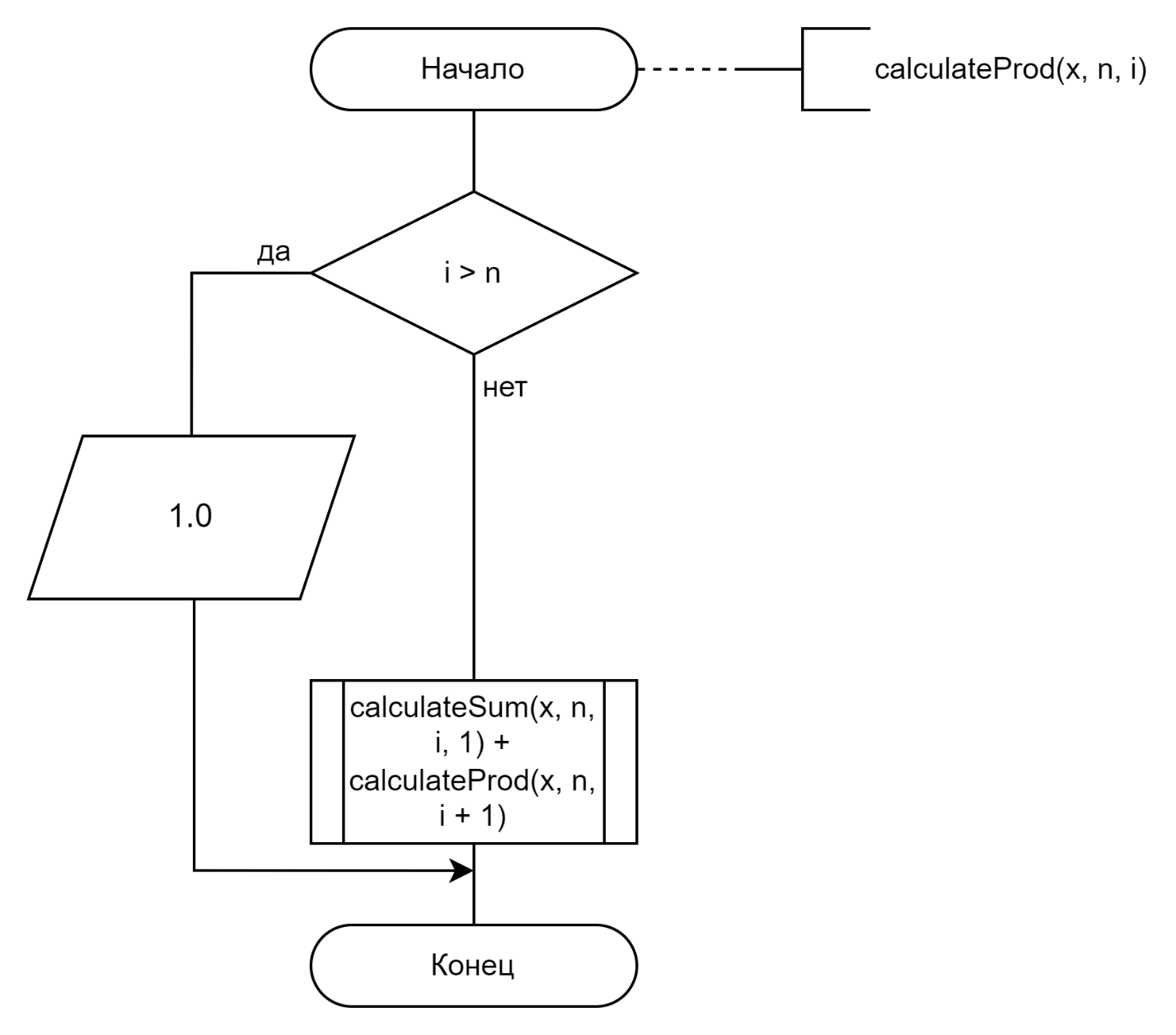


Рисунок 8 – Алгоритм функции calculateProd

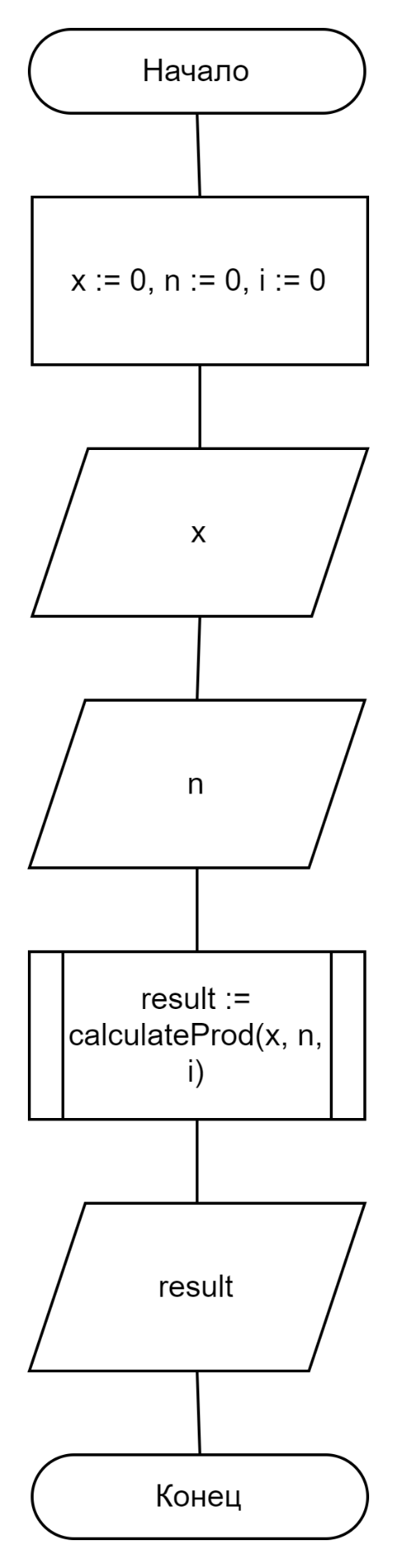


Рисунок 9 – Алгоритм решения задачи

***Текст программы***

*Текст программы для решения задачи 1*

// TODO: Описать бестиповую функцию Time24ToTime12 (H, M, S, am), которая преобразует время, заданное в часах H, минутах M и секундах S (H, M и S — целые положительные) из 24-часового формата в 12-часовой, при этом для времени в 12-часовом формате устанавливается признак принадлежности момента времени первой половине суток am (am - логическое). Используя эту функцию, привести к новому формату два заданных временных отрезка: (T1, M1, S1) и (T2, M2, S2).

#include <iostream>

using namespace std;

void Time24ToTime12(int\* H, int\* M, int\* S, bool\* am) {

// Проверка на корректность входных данных

if (H == nullptr || M == nullptr || S == nullptr || am == nullptr) {

cout << "Ошибка: Нулевой указатель." << endl;

return;

}

// Устанавливаем признак принадлежности первой половине суток

\*am = (\*H < 12);

// Преобразуем часы в 12-часовой формат

\*H = (\*H % 12 == 0) ? 12 : \*H % 12;

}

int main() {

int T1 = 0, M1 = 0, S1 = 0;

int T2 = 0, M2 = 0, S2 = 0;

// Ввод значений

cin >> T1 >> M1 >> S1;

cin >> T2 >> M2 >> S2;

bool am1, am2;

// Преобразование первого временного отрезка

int\* pT1 = &T1;

int\* pM1 = &M1;

int\* pS1 = &S1;

bool\* pam1 = &am1;

Time24ToTime12(pT1, pM1, pS1, pam1);

// Преобразование второго временного отрезка

int\* pT2 = &T2;

int\* pM2 = &M2;

int\* pS2 = &S2;

bool\* pam2 = &am2;

Time24ToTime12(pT2, pM2, pS2, pam2);

// Вывод результатов

cout << "fst time interval: " << T1 << ":" << M1 << ":" << S1;

if (am1) {

cout << " AM" << endl;

} else {

cout << " PM" << endl;

}

cout << "snd time interval: " << T2 << ":" << M2 << ":" << S2;

if (am2) {

cout << " AM" << endl;

} else {

cout << " PM" << endl;

}

}

*Текст программы для решения задачи 2*

// TODO: Даны два массива целых чисел размеров n и m и целые числа k и t. Если в первом массиве все элементы не превосходят k, а во втором массиве все элементы не превосходят t, то в первом массиве заменить все элементы равные k значением t, в противном случае во втором массиве заменить все элементы, равные t значением k.

#include <iostream>

using namespace std;

bool check(int\* mas, int k){

bool condition = true;

for (int i = 0; i < sizeof(mas); i++){

if (mas[i] > k){

condition = false;

break;

}

}

return condition;

}

void repl(int\* mas, int k, int t){

if (check(mas, k)){

for(int i = 0; i < sizeof(mas); i++){

if(mas[i] == k){

mas[i] = t;

}

}

}

else{

for(int i = 0; i < sizeof(mas); i++){

if(mas[i] == t){

mas[i] = k;

}

}

}

}

int main(){

int n, m;

cout << "Enter lenght of arrays: " << endl;

cin >> n >> m;

int\* arr1 = new int[n];

int\* arr2 = new int[m];

int k, t;

cout << "Enter 'k' and 't' elements: " << endl;

cin >> k >> t;

cout << "Enter the arrays elements for arr1:" << endl;

for (int i = 0; i < n; i++){

cin >> arr1[i];

}

cout << "Enter the arrays elements for arr2:" << endl;

for (int i = 0; i < m; i++){

cin >> arr2[i];

}

repl(arr1, k, t);

repl(arr2, k, t);

cout << "Result of work: " << endl;

for (int i = 0; i < n; i++){

cout << arr1[i] << " ";

}

cout << endl;

for (int i = 0; i < m; i++){

cout << arr2[i] << " ";

}

}

*Текст программы для решения задачи 3*

#include <iostream>

using namespace std;

double f(double x, int i, int j){

return x \* j \* 2 + i;

}

double calculateSum(double x, int n, int i, int j){

if (j > n){

return 0.0;

}

return f(x, i, j) + calculateSum(x, n, i, j + 1);

}

double calculateProd(double x, int n, int i){

if (i > n){

return 1.0;

}

return calculateSum(x, n, i, 1) + calculateProd(x, n, i + 1);

}

int main(){

double x = 0;

int n = 0;

int i = 0;

cout << "Enter x: ";

cin >> x;

cout << "Enter n: ";

cin >> n;

cout << "Result of programm: " << endl;

double result = calculateProd(x, n, i);

cout << result;

}

***Тестирование программы***

Тестирование задачи 1 представлено на рисунках 10, 11.

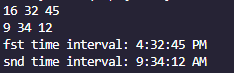


Рисунок 10 –Тест 1 задачи 1

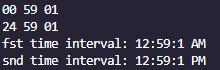


Рисунок 11 – Тест 2 задачи 1

Тестирование задачи 2 представлено на рисунках 12, 13.

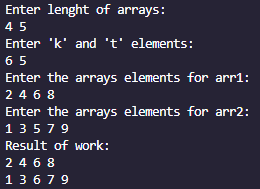


Рисунок 12 –Тест 1 задачи 2

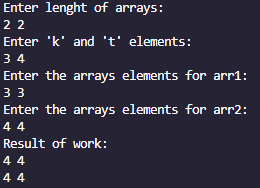


Рисунок 13 –Тест 2 задачи 2

Тестирование задачи 2 представлено на рисунках 14, 15.

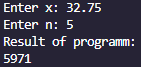


Рисунок 14 – Тест 1 Задачи 3

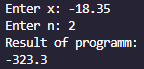


Рисунок 15 – Тест 2 Задачи 3