УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе №2.3

по предмету

Основы алгоритмизации и программирования

Вариант 3

Выполнил:

Бетеня К.С.

Проверила:

Данилова Г.В.

Группа 351005

Минск 2023

Задание:

Разработать алгоритмы и программы, реализующие эти алгоритмы. Основные функции программ оформить в виде процедур и функций. Исходные данные вводить из текстового файла. Результаты работы программы также поместить в текстовый файл и вывести на экран.

Дано натуральное n-значное число P. Проверить, является ли данное число палиндромом (перевертышем).

Код программы на **Delphi**:

Program labt3;

{$APPTYPE CONSOLE}

{$R \*.res}

Uses

System.SysUtils;

Const

CONS\_NUM: Integer = 1;

FILE\_NUM: Integer = 2;

PALIN\_OUTPUT\_CONTROL: Integer = -1;

Procedure PathCondition(Var Num: Integer; Var IsCorrect: Boolean);

Begin

Try

Readln(Num);

Except

Writeln('Invalid numeric input. Try again.');

End;

If (Num <> CONS\_NUM) And (Num <> FILE\_NUM) Then

Writeln('Choose only ', CONS\_NUM, ' or ', FILE\_NUM, '. Try again.')

Else

IsCorrect := True;

End;

Function ChoosingAPath(): Integer;

Var

Num: Integer;

IsCorrect: Boolean;

Begin

IsCorrect := False;

Repeat

Write('Your choice: ');

PathCondition(Num, IsCorrect);

Until IsCorrect;

ChoosingAPath := Num;

End;

Procedure PalinCondition(Var IsCorrect: Boolean; Var Palindrome: Integer);

Begin

Try

Readln(Palindrome);

Except

Writeln('Invalid numeric input. Try again.');

End;

If Palindrome < 1 Then

Writeln('Number should be natural.')

Else

IsCorrect := True;

End;

Function InputPalin(): Integer;

Var

Palindrome: Integer;

IsCorrect: Boolean;

Begin

IsCorrect := False;

Repeat

Write('Write your number: ');

PalinCondition(IsCorrect, Palindrome);

Until IsCorrect;

InputPalin := Palindrome;

End;

Function LengthOfPalin(Palindrome: Integer): Integer;

Var

PalinLen: Integer;

Begin

PalinLen := 0;

While (Palindrome > 0) Do

Begin

Inc(PalinLen);

Palindrome := Palindrome Div 10;

End;

LengthOfPalin := PalinLen;

End;

Procedure PutInMassive(Var ArrPalin: Array Of Integer; Palindrome: Integer);

Var

I: Integer;

Begin

I := 0;

While Palindrome > 0 Do

Begin

ArrPalin[I] := Palindrome Mod 10;

Inc(I);

Palindrome := Palindrome Div 10;

End;

End;

Function PalinIsPalin(Var ArrPalin: Array Of Integer; PalinLen: Integer; Palindrome: Integer): Boolean;

Var

IsCorrect: Boolean;

I: Integer;

Begin

IsCorrect := True;

For I := 0 To PalinLen Div 2 Do

If (ArrPalin[I] <> ArrPalin[PalinLen - I - 1]) Then

IsCorrect := False;

If Palindrome < 0 Then

IsCorrect := False;

PalinIsPalin := IsCorrect;

End;

Function PalinCheack(Palindrome: Integer): Boolean;

Var

PalinLen: Integer;

ArrPalin: Array Of Integer;

Begin

PalinLen := LengthOfPalin(Abs(Palindrome));

SetLength(ArrPalin, PalinLen);

PutInMassive(ArrPalin, Abs(Palindrome));

PalinCheack := PalinIsPalin(ArrPalin, PalinLen, Palindrome);

End;

Procedure ViaConsole();

Var

Palindrome: Integer;

Begin

Palindrome := InputPalin();

If (PalinCheack(Palindrome) And (Palindrome > PALIN\_OUTPUT\_CONTROL)) Then

Write('It is palindrome.')

Else

Write('It is not a palindrome.');

End;

Procedure ConditionCheack(Sim: Char; IsCorrect: Boolean; Var Palindrome, N: Integer);

Begin

If ((Sim = '-') Or (Sim = '0')) And (IsCorrect = False) Then

Palindrome := PALIN\_OUTPUT\_CONTROL

Else

If ((Sim < '0') Or (Sim > '9')) Then

Palindrome := PALIN\_OUTPUT\_CONTROL

Else

Begin

Palindrome := Palindrome + (Ord(Sim) - 48) \* N;

N := N \* 10;

End;

End;

Procedure CheackForOneString(IsCorrect: Boolean; Var Palindrome: Integer);

Begin

If (IsCorrect = False) Then

Palindrome := PALIN\_OUTPUT\_CONTROL;

End;

Function InputPalinFile(Var MyFile: TextFile): Integer;

Var

Palindrome, N: Integer;

Sim: Char;

IsCorrect: Boolean;

Begin

Palindrome := 0;

N := 1;

IsCorrect := False;

Reset(MyFile);

Try

While Not EOF(MyFile) And (Palindrome <> PALIN\_OUTPUT\_CONTROL) Do

Begin

Read(MyFile, Sim);

ConditionCheack(Sim, IsCorrect, Palindrome, N);

IsCorrect := True;

End;

Finally

CloseFile(MyFile);

End;

CheackForOneString(IsCorrect, Palindrome);

InputPalinFile := Palindrome;

End;

Procedure OutputPalin(Palindrome: Integer; Var MyFile: TextFile);

Begin

Append(MyFile);

If Palindrome = PALIN\_OUTPUT\_CONTROL Then

Write(MyFile, #13#10, 'ERROR.')

Else

If (PalinCheack(Palindrome)) Then

Write(MyFile, #13#10, 'It is palindrome.')

Else

If (Palindrome <> PALIN\_OUTPUT\_CONTROL) Then

Write(MyFile, #13#10, 'It is not a palindrome.');

CloseFile(MyFile);

End;

Procedure WorkWithFile(Var MyFile: TextFile);

Var

Palindrome: Integer;

Begin

Palindrome := InputPalinFile(MyFile);

OutputPalin(Palindrome, MyFile);

Write('Cheack your file.');

End;

Procedure WayCondition(Way: String; Var IsCorrect: Boolean);

Var

Bufstr: String;

Begin

If Way.Length > 4 Then

Begin

Bufstr := Way.Substring(Way.Length - 4);

If Bufstr = '.txt' Then

IsCorrect := True

Else

Writeln('Write .txt file.');

End

Else

Writeln('The path is too short.');

End;

Function InputWay(): String;

Var

Way: String;

IsCorrect: Boolean;

Begin

IsCorrect := False;

Repeat

Read(Way);

WayCondition(Way, IsCorrect);

Until IsCorrect;

InputWay := Way;

End;

Procedure ViaFile();

Var

FileWay: String;

MyFIle: TextFile;

Begin

Write('Write way to your file: ');

FileWay := InputWay();

Try

AssignFile(MyFile, FileWay);

Reset(MyFile);

WorkWithFile(MyFile);

Except

Write('Bad File.');

End;

End;

Var

Option: Integer;

Resoult: String;

Begin

Writeln('The program determines whether', #13#10#9, 'the entered natural

number is a palindrome.', #13#10);

Writeln('Where will we work through: ', #13#10#9, 'Console: ', CONS\_NUM, #9,

'File: ', FILE\_NUM, #13#10);

Option := ChoosingAPath();

If Option = FILE\_NUM Then

ViaFile()

Else

ViaConsole();

Readln;

Readln;

End.

Код программы на **C++**:

#include <iostream>

#include <fstream>

using namespace std;

const int CONS\_NUM = 1;

const int FILE\_NUM = 2;

const int PALIN\_OUTPUT\_CONTROL = -1;

void wayCondition(string way, bool& isIncorrect) {

if (way.size() > 4) {

string bufstr = way.substr(way.size() - 4);

if (bufstr == ".txt")

isIncorrect = false;

else

cout << "Write .txt file.\n";

}

else

cout << "The path is too short.\n";

}

string inputWay() {

string way;

bool isIncorrect = true;

do {

cin >> way;

wayCondition(way, isIncorrect);

} while (isIncorrect);

return way;

}

void pathCondition(int& num, bool& isIncorrect) {

cin >> num;

if (cin.fail() || cin.get() != '\n') {

cout << "Invalid numeric input. Try again.\n";

cin.clear();

while (cin.get() != '\n');

}

else if (num != CONS\_NUM && num != FILE\_NUM)

cout << "Choose only " << CONS\_NUM << " or "

<< FILE\_NUM << ". Try again.\n";

else

isIncorrect = false;

}

int choosingAPath() {

int num;

bool isIncorrect = true;

do {

cout << "Your choice: ";

pathCondition(num, isIncorrect);

} while (isIncorrect);

return num;

}

void palinCondition(bool& isIncorrect, int& palindrome) {

cin >> palindrome;

if (cin.fail() || cin.get() != '\n') {

cout << "Invalid numeric input.Try again.\n";

cin.clear();

while (cin.get() != '\n');

}

else if (palindrome < 1)

cout << "Number should be natural.\n";

else

isIncorrect = false;

}

int inputPalin() {

bool isIncorrect = true;

int palindrome;

do {

cout << "Write your number: ";

palinCondition(isIncorrect, palindrome);

} while (isIncorrect);

return palindrome;

}

void conditionCheack(char sim, bool isCorrect, int& palindrome, int& n) {

if ((sim == '-' || sim == '0') && !isCorrect)

palindrome = PALIN\_OUTPUT\_CONTROL;

else if (sim < '0' || sim > '9')

palindrome = PALIN\_OUTPUT\_CONTROL;

else {

palindrome += (sim - 48) \* n;

n \*= 10;

}

}

void cheackForOneString(bool isCorrect, int& palindrome) {

if (!isCorrect)

palindrome = PALIN\_OUTPUT\_CONTROL;

}

int inputPalinFile(fstream& file) {

int palindrome = 0, n = 1;

char sim;

bool isCorrect = false;

while (file.get(sim) && palindrome != PALIN\_OUTPUT\_CONTROL) {

conditionCheack(sim, isCorrect, palindrome, n);

isCorrect = true;

}

cheackForOneString(isCorrect, palindrome);

return palindrome;

}

int lengthOfPalin(int palindrome) {

int palinLen = 0;

while (palindrome) {

palinLen++;

palindrome /= 10;

}

return palinLen;

}

void putInMassive(int\*& arrPalin, int palindrome) {

int i = 0;

while (palindrome) {

arrPalin[i] = palindrome % 10;

i++;

palindrome = palindrome / 10;

}

}

bool palinIsPalin(int\*& arrPalin, int palinLen, int palindrome) {

bool isCorrect = true;

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

if (arrPalin[i] != arrPalin[palinLen - i - 1])

isCorrect = false;

}

if (palindrome < 0)

isCorrect = false;

return isCorrect;

}

bool palinCheack(int palindrome) {

int palinLen = lengthOfPalin(palindrome);

int\* arrPalin = new int[palinLen];

putInMassive(arrPalin, palindrome);

return palinIsPalin(arrPalin, palinLen, palindrome);

}

void viaConsole() {

int palindrome = inputPalin();

if (palinCheack(palindrome) && palindrome > PALIN\_OUTPUT\_CONTROL)

cout << "It is palindrome.";

else

cout << "It is not a palindrome.";

}

void outputPalin(int palindrome, fstream& file) {

file.seekg(0, ios::end);

if (palindrome == PALIN\_OUTPUT\_CONTROL)

file << "\nERROR.";

else if (palinCheack(palindrome))

file << "\nIt is palindrome.";

else if (palindrome != PALIN\_OUTPUT\_CONTROL)

file << "\nIt is not a palindrome.";

}

void workWithFile(fstream& file) {

if (file.is\_open()) {

int palindrome = inputPalinFile(file);

file.clear();

outputPalin(palindrome, file);

cout << "Cheack your file.";

}

else

cout << "Bad File.";

}

void viaFile() {

string fileWay;

cout << "Write way to your file: ";

fileWay = inputWay();

fstream file;

file.open(fileWay);

workWithFile(file);

file.close();

}

int main() {

cout << "The program determines whether\n\t"

<< "the entered natural number is a palindrome.\n\n";

cout << "Where will we work through: \n\tConsole: "

<< CONS\_NUM << "\tFile: " << FILE\_NUM << "\n\n";

int option = choosingAPath();

option == FILE\_NUM ? viaFile() : viaConsole();

return 0;

}

Код программы на **Java**:

package lab2;  
  
import java.io.\*;  
import java.util.Scanner;  
import java.util.concurrent.atomic.AtomicBoolean;  
import java.util.concurrent.atomic.AtomicInteger;  
  
public class lab3 {  
 static final int CONS\_NUM = 1;  
 static final int FILE\_NUM = 2;  
 static final int PALIN\_OUTPUT\_CONTROL = -1;  
  
 static void wayCondition(String way, AtomicBoolean isIncorrect) {  
 if (way.length() > 4) {  
 String bufstr = way.substring(way.length() - 4);  
 if (bufstr.equals(".txt"))  
 isIncorrect.set(false);  
 else  
 System.out.print("Write .txt file.\n");  
 }  
 else  
 System.out.print("The path is too short.\n");  
 }  
  
  
 static String inputWay(Scanner in) {  
 String way;  
 AtomicBoolean isIncorrect = new AtomicBoolean(true);  
 do {  
 way = in.nextLine();  
 wayCondition(way, isIncorrect);  
 } while (isIncorrect.get());  
  
 return way;  
 }  
  
 static void pathCondition(AtomicInteger num, AtomicBoolean isIncorrect,

Scanner in) {  
 try {  
 num.set(Integer.parseInt(in.nextLine()));  
 } catch (NumberFormatException error) {  
 System.out.print("Invalid numeric input. Try again.\n");  
 }  
 if (num.get() != CONS\_NUM && num.get() != FILE\_NUM)  
 System.out.printf("Choose only %d or %d. Try again.\n", CONS\_NUM,

FILE\_NUM);  
 else isIncorrect.set(false);  
 }  
  
 static int choosingAPath(Scanner in) {  
 AtomicInteger num = new AtomicInteger(0);  
 AtomicBoolean isIncorrect = new AtomicBoolean(true);  
 do {  
 System.out.print("Your choice: ");  
 pathCondition(num, isIncorrect, in);  
 } while (isIncorrect.get());  
  
 return num.get();  
 }  
  
 static void palinCondition(AtomicBoolean isIncorrect, AtomicInteger

palindrome, Scanner in) {  
 try{  
 palindrome.set(Integer.parseInt(in.nextLine()));  
 } catch (NumberFormatException error) {  
 System.out.print("Invalid numeric input.Try again.\n");  
 }  
 if (palindrome.get() < 1)  
 System.out.print("Number should be natural.\n");  
 else  
 isIncorrect.set(false);  
 }  
  
  
 static int inputPalin(Scanner in) {  
 AtomicBoolean isIncorrect = new AtomicBoolean(true);  
 AtomicInteger palindrome = new AtomicInteger(0);  
 do {  
 System.out.print("Write your number: ");  
 palinCondition(isIncorrect, palindrome, in);  
 } while (isIncorrect.get());  
  
 return palindrome.get();  
 }  
  
  
 static int lengthOfPalin(int palindrome) {  
 int palinLen = 0;  
 while (palindrome > 0) {  
 palinLen++;  
 palindrome /= 10;  
 }  
  
 return palinLen;  
 }  
  
  
 static void putInMassive(int[] arrPalin, int palindrome) {  
 int i = 0;  
 while (palindrome > 0) {  
 arrPalin[i] = palindrome % 10;  
 i++;  
 palindrome = palindrome / 10;  
 }  
 }  
  
  
 static boolean palinIsPalin(int[] arrPalin, int palinLen, int palindrome) {  
 boolean isCorrect = true;  
 for (int i = 0; i < palinLen / 2; i++) {  
 if (arrPalin[i] != arrPalin[palinLen - i - 1])  
 isCorrect = false;  
 }  
 if (palindrome < 0)  
 isCorrect = false;  
  
 return isCorrect;  
 }  
  
  
 static boolean palinCheack(int palindrome) {  
 int palinLen = lengthOfPalin(palindrome);  
 int[] arrPalin = new int[palinLen];  
 putInMassive(arrPalin, palindrome);  
  
 return palinIsPalin(arrPalin, palinLen, palindrome);  
 }  
  
  
 static void viaConsole(Scanner in) {  
 int palindrome = inputPalin(in);  
 if (palinCheack(palindrome) && palindrome > PALIN\_OUTPUT\_CONTROL)  
 System.out.print("It is palindrome.");  
 else  
 System.out.print("It is not a palindrome.");  
 }  
  
  
 static void conditionCheack(char sim, boolean isCorrect, AtomicInteger

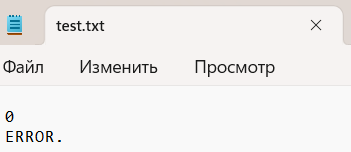
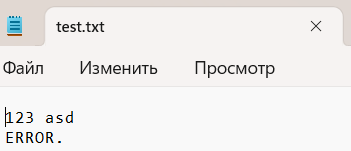
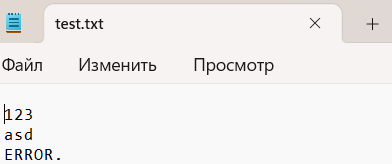
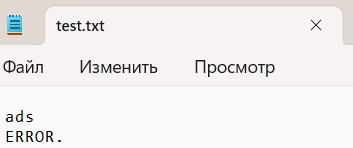
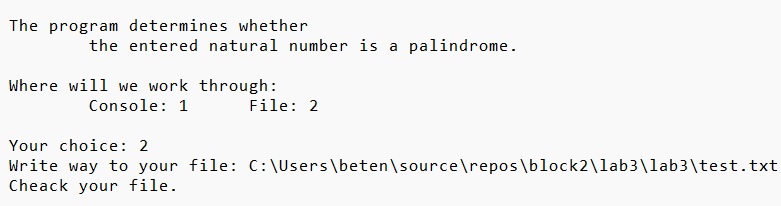
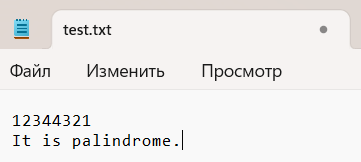
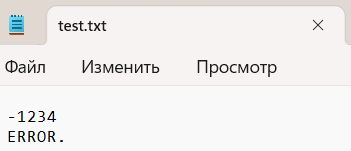
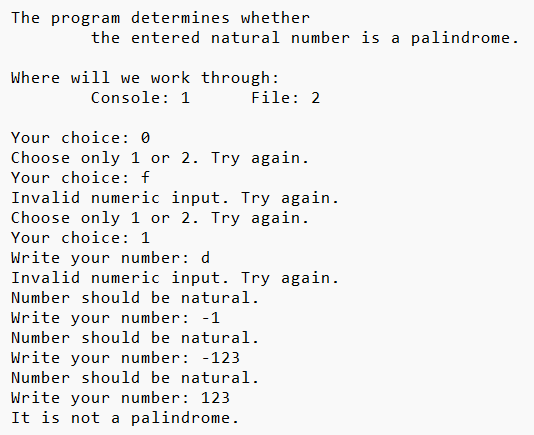
palindrome, AtomicInteger n) {  
 if ((sim == '-' || sim == '0') && !isCorrect)  
 palindrome.set(PALIN\_OUTPUT\_CONTROL);  
 else if (sim < '0' || sim > '9')  
 palindrome.set(PALIN\_OUTPUT\_CONTROL);  
 else {  
 palindrome.set(palindrome.get() + (sim - 48) \* n.get());  
 n.set(n.get() \* 10);  
 }  
 }  
  
  
 static void cheackForOneString(boolean isCorrect, AtomicInteger palindrome) {  
 if (!isCorrect)  
 palindrome.set(PALIN\_OUTPUT\_CONTROL);  
 }  
  
 static int inputPalinFile(FileReader fileReader) throws IOException {  
 AtomicInteger palindrome = new AtomicInteger(0);  
 AtomicInteger n = new AtomicInteger(1);  
 int cs;  
 char sim;  
 boolean isCorrect = false;  
 while ((cs = fileReader.read()) != -1 && palindrome.get() !=

PALIN\_OUTPUT\_CONTROL) {  
 sim = (char) cs;  
 conditionCheack(sim, isCorrect, palindrome, n);  
 isCorrect = true;  
 }  
 cheackForOneString(isCorrect, palindrome);  
  
 return palindrome.get();  
 }  
  
 static void outputPalin(int palindrome,File fileWay) throws IOException {  
 FileWriter writer = new FileWriter(fileWay, true);  
 if (palindrome == PALIN\_OUTPUT\_CONTROL) {  
 writer.write("\nERROR.");  
 }  
 else if (palinCheack(palindrome))  
 writer.write("\nIt is palindrome.");  
 else writer.write("\nIt is not a palindrome.");  
 writer.close();  
 }  
  
 static void workWithFile(String fileWay, Scanner in){  
 assert fileWay != null;  
 File file = new File(fileWay);  
 try {  
 FileReader fileReader = new FileReader(file);  
 BufferedReader bufferedReader = new BufferedReader(fileReader);  
 int palindrome = inputPalinFile(fileReader);  
 bufferedReader.close();  
 outputPalin(palindrome, new File(fileWay));  
 System.out.print("Cheack your file.");  
 } catch (IOException error) {  
 System.err.println("Bad File.");  
 }  
 }  
  
  
 static void viaFile(Scanner in) {  
 String fileWay;  
 System.out.print("Write way to your file: ");  
 fileWay = inputWay(in);  
 workWithFile(fileWay, in);  
 }  
  
  
 public static void main(String[] args) {  
 Scanner in = new Scanner(System.in);  
 System.out.print("The program determines whether\n\tthe entered natural

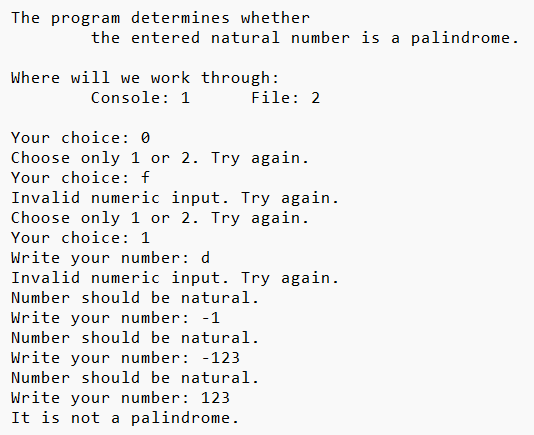
number is a palindrome.\n\n");  
 System.out.printf("Where will we work through: \n\tConsole: %d \tFile:

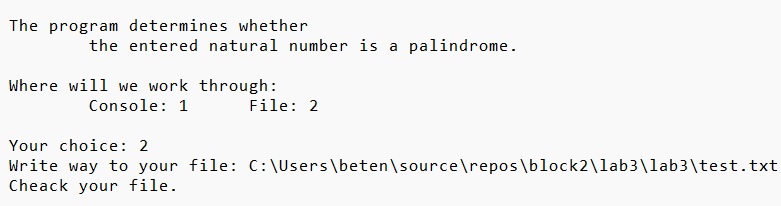
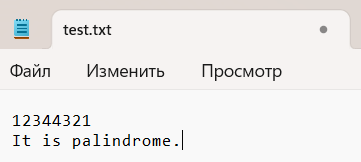
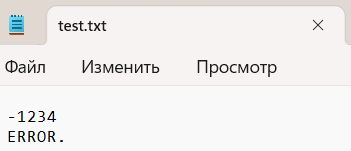
%d\n\n", CONS\_NUM, FILE\_NUM);  
 int option = choosingAPath(in);  
  
 if (option == FILE\_NUM) {  
 viaFile(in);  
 } else {  
 viaConsole(in);  
 }  
  
 in.close();  
 }  
}

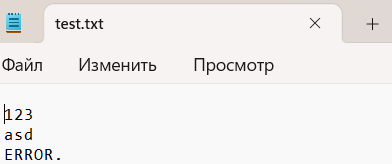
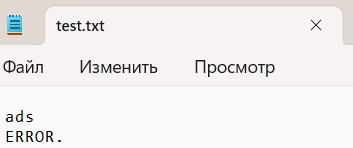
Результат на **Delphi**:

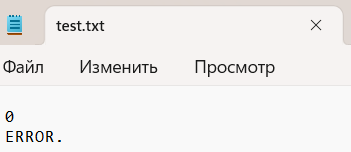
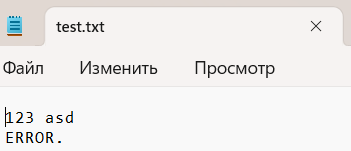


Результат на **Java**:

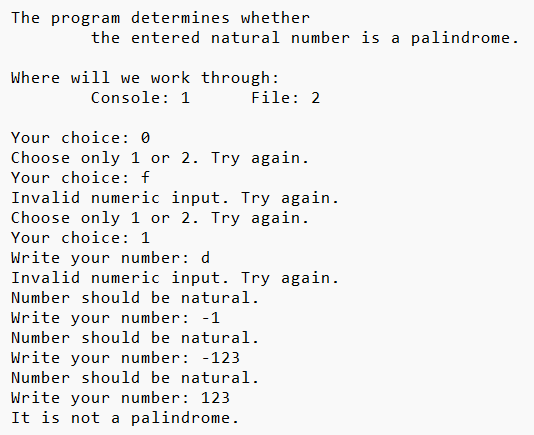


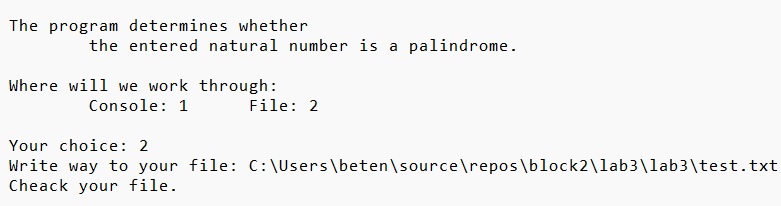
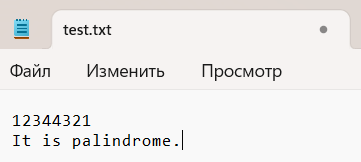
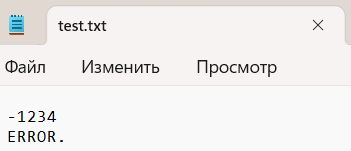


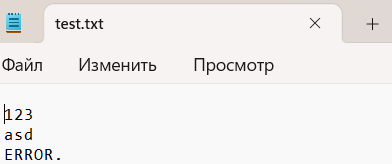
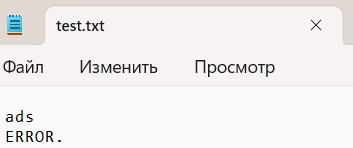


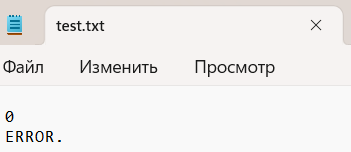
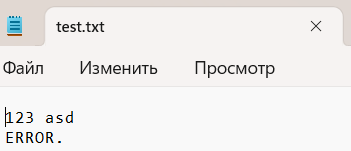


Результат на **C++**:









Блок-схема:

