1. **ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ**
   1. **Объект испытаний**

Наименование темы разработки – «Гармонический синтез».

Наименование испытуемой программы – «Гармонический синтез.exe».

Программой будут пользоваться те люди, которым необходимо быстро разложить сигнал на гармонические составляющие.

* 1. **Цель испытаний**

Проверка соответствия основных характеристик программы функциональным и иным видам требований, изложенным в программном документе «Техническое задание», является целью испытаний.

* 1. **Требования к программе**

Требования к программе описаны в п. «Требования к функциональным характеристикам» Технического задания.

* 1. **Требования к программной документации**

Состав программной документации должен включать в себя:

1) техническое задание;

2) описание программы;

3) программу и методики испытаний;

4) пояснительную записку;

5) ведомость эксплуатационных документов;

6) описание применения;

7) руководство программиста;

8) руководство оператора.

* 1. **Средства и порядок испытаний**

ПК с ОЗУ более 32 Мбайт, 8 МБ видеопамяти и выше, наличием более 30 Мбайт свободного места на жестком диске.

Проверки, проводимые во время испытаний:

а) проверка комплектности программной документации;

б) проверка комплектности и состава технических и программных средств;

в) проверка соответствия технических характеристик программы;

г) проверка степени выполнения требований функционального назначения программы.

* 1. **Методы испытаний**

Проверка комплектности программной документации на программное изделие производится визуально представителем службы, ответственной за эксплуатацию. В ходе проверки сопоставляется состав и комплектность программной документации, представленной Разработчиком, с перечнем программной документации, приведенным в п. «Состав программной документации, предъявляемой на испытания» настоящего документа. Проверка считается завершенной в случае соответствия состава и комплектности программной документации, представленной Разработчиком, перечню программной документации, приведенному в указанном выше пункте. По результатам проведения проверки, представитель службы, ответственной за эксплуатацию вносит запись в Протокол испытаний – “Комплектность программной документации соответствует (не соответствует) требованиям п. «Состав программной документации, предъявляемой на испытания»” настоящего документа.

Проверка комплектности и состава технических и программных средств производится визуально представителем службы, ответственной за эксплуатацию. В ходе проверки сопоставляется состав и комплектность технических и программных средств, представленных Разработчиком, с перечнем технических и программных средств, приведенным в п. «Технические средства, используемые во время испытаний» и п. «Программные средства, используемые во время испытаний» настоящего документа. Комплектность программных средств проводится также визуально. Загрузилась операционная система, высветился логотип, версия - соответствует/не соответствует заявленной в Техническом задании и т.д. Проверка считается завершенной в случае соответствия состава и комплектности технических и программных средств, представленных Разработчиком, с перечнем технических и программных средств, приведенных в пп. «Технические средства, используемые во время испытаний» и «Программные средства, используемые во время испытаний» настоящего документа. По результатам проведения проверки представитель службы, ответственной за эксплуатацию, вносит запись в Протокол испытаний - «Комплектность технических и программных средств соответствует (не соответствует) требованиям пп. «Технические средства, используемые во время испытаний» и «Программные средства, используемые во время испытаний» настоящего документа».

Проверка работоспособности программы выполняется согласно п. «Проверка работоспособности программы» Руководства системного программиста. Проверка считается завершенной в случае соответствия состава и последовательности действий, при выполнении данной проверки, указанному выше подразделу Руководства системного программиста. По результатам проведения проверки представитель службы, ответственной за эксплуатацию вносит запись в Протокол испытаний - «п. “Проверка работоспособности программы“ выполнена».

* 1. **Проверка работоспособности программы**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назначение проверки | Номер теста | Значения входных переменных | Ожидаемый результат работы программы | Результат тестирования |
| Проверка работоспособности в рабочем диапазоне данных | 1 | 2, 3, 4, 1, 2, 56, 87 | Программа работает корректно.  Расчет происходит. | Рисунок 1 Работа программы с числами внутри рабочего диапазона |
| 2 | чтение | Программа работает корректно.  Чтение из файла происходит. | Рисунок 2 Чтение из файла |
| 3 | Очистка,  чтение | Программа работает корректно.  Чтение из не файла происходит. | Рисунок 3 Очистка файла и последующее чтение |
| Проверка реакции программы на ввод данных вне рабочего диапазона | 4 | -4 | Программа работает корректно.  Присвоение переменной значения не происходит. | Рисунок 4 Работа программы с числами вне рабочего диапазона |
| Проверка реакции на ошибку типа входных данных | 5 | QQ | Программа работает корректно.  Присвоение переменной значения не происходит. | Рисунок 5 Работа программы с неверными типами данных |

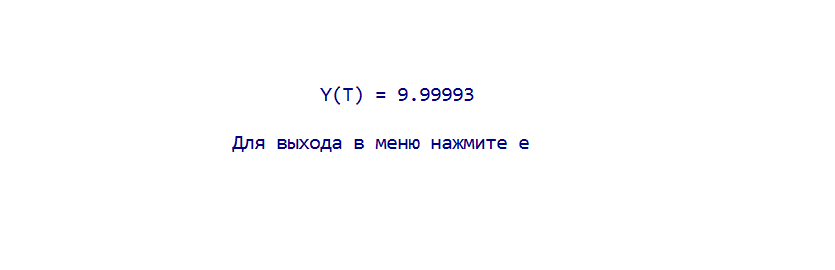


Рисунок 1 Работа программы с числами внутри рабочего диапазона

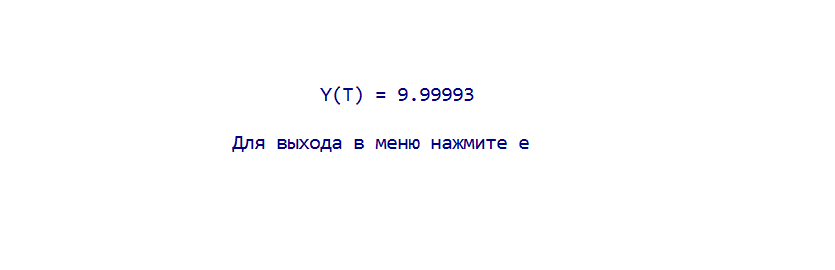


Рисунок 2 Чтение из файла

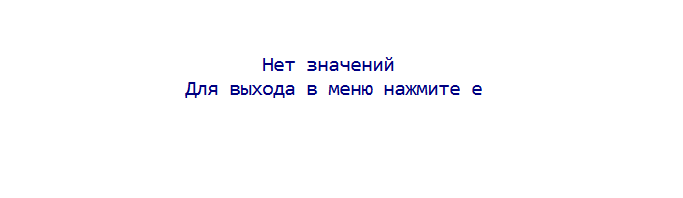


Рисунок 3 Очистка файла и последующее чтение

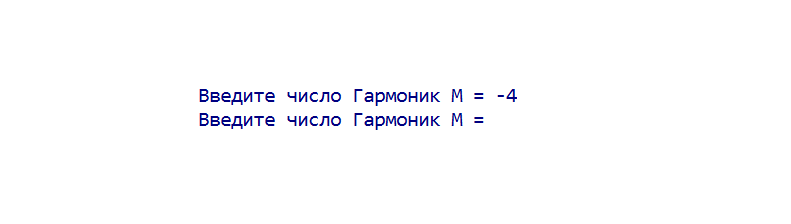


Рисунок 4 Работа программы с числами вне рабочего диапазона

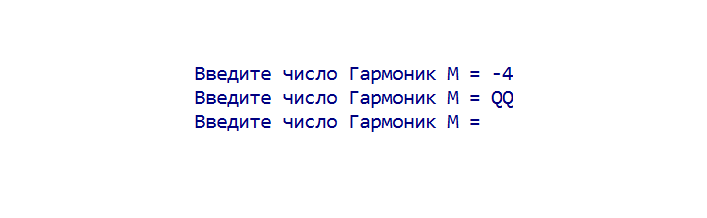


Рисунок 5 Работа программы с неверными типами данных

* 1. **Текст программы**

#include <iostream>

#include <math.h>

#include <clocale>

#include <iomanip>

#include <Windows.h>

#include <conio.h>

#include <fstream>

using namespace std;

//работа с файлами

ofstream out;

ifstream in;

// переменные

int M, F, T, i, sum, action2;

int \* A;

int \* B;

double C, S, D, Y;

const double PI = 3.14159265;

const string filename = "znach.txt";

// прототипы функций

int math();

void space(int n);

bool checkIsDigit(char \*str);

void readFromFile();

void clearFile();

void reference();

int main() {

system("color F1");

setlocale(LC\_ALL, "Russian");

do

{

space(10);

cout << setw(72) << "ГАРМОНИЧЕСКИЙ СИНТЕЗ \n\n\n";

cout << setw(76) << "Для получения справки нажмите a \n";

cout << setw(75) << "Для начала расчета нажмите b \n";

cout << setw(80) << "Для считывания данных из файла нажмите c \n";

cout << setw(74) << "Для очистки файла нажмите d \n";

cout << setw(71) << "Для выхода нажмите e \n";

int action = \_getch();

switch (action)

{

case 97: {

reference();

break;

}

case 98: {

math();

break;

}

case 99:

{

readFromFile();

break;

}

case 100: {

clearFile();

break;

}

case 101:

{

space(10);

cout << setw(72) << "Программа завершена";

Sleep(2000);

return 0;

}

default:

{

space(10);

cout << setw(67) << "Ошибка";

Sleep(1500);

space(10);

}

}

} while (1);

}

int math()

{

space(10);

do {

cout << setw(70) << "Введите число Гармоник M = ";

char num\_ch[100];

cin >> num\_ch;

if (checkIsDigit(num\_ch))

M = atoi(num\_ch);

else M = -1;

} while (M < 0);

A = new int[M];

B = new int[M];

for (i = 0; i < M; i++) {

space(10);

do {

cout << setw(62) << "Введите А [" << i << "] = ";

char num\_ch[100];

cin >> num\_ch;

if (checkIsDigit(num\_ch))

A[i] = atoi(num\_ch);

else A[i] = -1;

} while (A[i] < 0);

}

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

space(10);

do {

cout << setw(62) << "Введите B [" << i << "] = ";

char num\_ch[100];

cin >> num\_ch;

if (checkIsDigit(num\_ch))

B[i] = atoi(num\_ch);

else B[i] = -1;

} while (B[i] < 0);

}

space(10);

do {

cout << setw(68) << "Введите частоту F = ";

char num\_ch[100];

cin >> num\_ch;

if (checkIsDigit(num\_ch))

F = atoi(num\_ch);

else F = -1;

} while (F < 0);

space(10);

do {

cout << setw(67) << "Введите время T = ";

char num\_ch[100];

cin >> num\_ch;

if (checkIsDigit(num\_ch))

T = atoi(num\_ch);

else T = -1;

} while (T < 0);

C = 2 \* PI\*F\*T;

S = 0;

for (i = 0; i < M; i++) {

D = C\*i;

S = S + (A[i] \* cos(D) + B[i] \* sin(D));

sum = sum + A[i];

}

Y = sum / 2 + S;

space(10);

cout << setw(63) << "Y(T) = " << Y << "\n\n";

out.open(filename);

out << Y;

out.close();

do {

cout << setw(77) << "Для выхода в меню нажмите e \n";

action2 = \_getch();

} while (action2 != 101);

return 0;

}

void space(int n) {

system("cls");

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

cout << endl;

}

bool checkIsDigit(char \*str)

{

while (\*str != 0)

{

if (!isdigit(\*str++))

return false;

}

return true;

}

void readFromFile()

{

space(10);

in.open(filename);

in >> Y;

in.close();

if (Y == NULL)

cout << setw(69) << "Нет значений \n";

else

cout << setw(63) << "Y(T) = " << Y << "\n\n";

do {

cout << setw(77) << "Для выхода в меню нажмите e \n";

action2 = \_getch();

} while (action2 != 101);

}

void clearFile()

{

Y = NULL;

out.open(filename);

out << Y;

out.close();

}

void reference()

{

space(10);

cout << setw(120) << "Гармонический синтез - получение колебаний сложной формы путем суммирования их гармонических составляющих (гармоник). \n" << setw(119) <<"Гармонический синтез может искать как самостоятельное значение, так и использоваться как средство тригонометрической \n"<< setw(75) << "интерполярной функций. \n\n";

do {

cout << setw(77) << "Для выхода в меню нажмите e \n";

action2 = \_getch();

} while (action2 != 101);

}