Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина) (СПбГЭТУ)

Кафедра МОЭВМ

**Лабораторная работа №6**

**“Организация связи Ассемблера с ЯВУ на примере программы**

**построения частотного распределение попаданий псевдослучайных целых чисел в заданные интервалы. ”**

По дисциплине «Архитектура компьютера»

Выполнила

Факультет КТИ

Группа № 4381

Преподаватель Кирьянчиков В.А.

Санкт Петербург

2015

# Постановка задачи

# На языке высокого уровня (Pascal или С) программируется ввод с клавиатуры и контроль исходных данных, а также генерируется массив псевдослучайных целых чисел, изменяющихся в заданном диапазоне и имеющих равномерное распределение.

# Далее должны вызываться 1 или 2 ассемблерных процедуры для формирования распределения количества попаданий псевдослучайных целых чисел в заданные интервалы. Ассемблерные процедуры должны вызываться как независимо скомпилированные модули. Передача параметров в процедуру должна выполняться через кадр стека.

# Результирующий массив частотного распределения чисел по интервалам, сформированный на ассемблерном уровне, возвращается в программу, реализованную на ЯВУ, и затем сохраняется в файле и выводится на экран средствами ЯВУ.

# Исходные данные:

# 1. Длина массива псевдослучайных целыхчисел - NumRanDat (<= 16K)

# 2. Диапазон изменения массива псевдослучайных целых чисел [Xmin, Xmax] (м.б. биполярный, например, [-100, 100])

# 3. Массив псевдослучайных целых чисел.

# 4. Количество интервалов, на которые разбивается диапазон изменения массива псевдослучайных целых чисел - NInt ( <=24 )

# 5. Массив левых границ интервалов разбиения LGrInt (должны принадлежать интервалу [Xmin,Xmax]) В общем случае интервалы разбиения диапазона изменения псевдослучайных чисел могут иметь различную длину.

# Результаты:

# Текстовая таблица, строка которой содержит:

# - номер интервала,

# - левую границу интервала,

# - количество псевдослучайных чисел, попавших в интервал.

# Количество строк должно быть равно числу интервалов разбиения.

# Таблица должна выводиться на экран и сохраняться в файле.

# 

# *Вариант 10*

# Программа формирования распределения количества попаданий псевдослучайных целых чисел в заданные интервалы реализуется в виде двух ассемблерных модулей (процедур), первый из которых формирует распределение исходных чисел по интервалам единичной длины и возвращает его в вызывающую программу на ЯВУ как промежуточный результат. Также это распределение выводится на экран и сохраняется в файле. Затем вызывается второй ассемблерный модуль, который по этому промежуточному распределению формирует окончательное распределение псевдослучайных целых чисел по интервалам произвольной длины ( с заданными границами ). Это распределение возвращается в головную программу и выдается как основной результат в виде текстового

# файла и, возможно, графика.

# Код программы.

#include <iostream>

#include <new>

#include <random>

#include <fstream>

int MAX\_NUMBER = 16384; //максимальное длина числа

int MAX\_COUNT\_BORD = 24; //максимальное количесво интервалов

bool GetInformation(int & NumRanDat, int\* & Number, int & Xmin, int & Xmax, int & NInt, int\* & LGrInt);//получение данных

bool InsertBorder(int\* & LGrInt,int cur\_len,int num,int min,int max);//вставка произвольных интервалов в массив

void GetRandomNum(int\* & Number,int len,int min,int max);//генерирование случайных чисел

void InitArray(int\* & Number,int len);//заполение массива 0-ми

void PrintResult1(int\* Number,int len);

void PrintResult2(int\* LGrInt ,int\* CountNum,int len)

extern "C"

{

void SORTSTEP1(int Number[], int NumRanDat,int CountNumUnit1[],int Xmin);

void SORTSTEP2(int CountNumUnit1[],int lenUnit1, int LGrInt[],int CountNumN[],int NInt,int Xmin);

}

int main(void)

{

int NumRanDat=0; //количество превдослучайных чисел

int \*Number=NULL; //массив чисел

int Xmin=0; //левая граница диапозона

int Xmax=0; //правая граница диапазона

int NInt=0; //количество границ (не в ключай граничный значения)

int \*LGrInt=NULL; //массив левых границ

int \*CountNumUnit1 =NULL;//для интеревалов еденичной длины

int lenUnit1=0;

int\* CountNumN=NULL; //для интервалов проивольной длины

if(!GetInformation(NumRanDat,Number,Xmin,Xmax,NInt,LGrInt)) //получить значения

return 1;

PrintArray(Number,NumRanDat); //вывести случайные числа

lenUnit1=Xmax-Xmin+1; //выделение памяти

CountNumUnit1= new int[lenUnit1];

if(!CountNumUnit1){ std::cout << "Ошибка: не удалось выделить память\n";return 1;}

InitArray(CountNumUnit1,lenUnit1);

CountNumN = new int[NInt];

if(!CountNumN){std::cout << "Ошибка: не удалось выделить память\n";return 1;}

InitArray(CountNumN,NInt);

SORTSTEP1(Number,NumRanDat,CountNumUnit1,Xmin); // распределение по интервалам 1

SORTSTEP2(CountNumUnit1,lenUnit1,LGrInt,CountNumN,NInt,Xmin); //распределение по инервалам произвольной длины

PrintResult1(CountNumUnit1,lenUnit1); //вывод результата первой процедуры

PrintResult2(LGrInt,CountNumN,NInt); //вывод результата второй процедуры

delete [] CountNumN;

delete [] CountNumUnit1;

delete [] Number;

delete [] LGrInt;

return 0;}

bool GetInformation(int & NumRanDat, int\* & Number, int & Xmin, int & Xmax, int & NInt, int\* & LGrInt)

{

int len=0;

int i=0;

setlocale(LC\_ALL, "rus");

//получение количества чисел

std::cout << "1.Введите количество случайных чисел, 0 < N < 16384: ";

std::cin >> NumRanDat;

std::cout << NumRanDat << std::endl;

while(NumRanDat <= 0 || NumRanDat >= 16384)

{ std::cin.clear();

std::cin.sync();

std::cout << "Ошибка: число должно быть в правильном диапазоне\n"

<<" Введите количество случайных чисел: ";

std::cin >> NumRanDat;}

//получение диапозона

std::cout << "2.Введите диапазон случайных чисел: \n"

<< "От: ";

std::cin >> Xmin;

std::cout << "До :";

std::cin >> Xmax;

while(Xmax <= Xmin)

{ std::cin.clear();

std::cin.sync();

std::cout << "Ошибка: правая граница диапозона должна быть больше левой\n"

<<"Введите правую границу: ";

std::cin >> Xmax;}

//выделяем память

Number=new int[NumRanDat];

if(!Number){std::cout << "Ошибка: не удалось выделить память\n";return false;}

GetRandomNum(Number,NumRanDat,Xmin,Xmax);

//получение количества интервалов

std::cout << "3.Введите количестно интервалов, но которые разобьется диапозон 0 < N < 25: ";

std::cin >> NInt;

while(NInt <= 0 || NInt >= 25)

{ std::cin.clear();

std::cin.sync();

std::cout << "Ошибка: количество интервалов не в указанном диапазоне\n"

<<"Введите количество интервалов: ";

std::cin >> NInt;}

LGrInt=new int[NInt];

if(!Number){ std::cout << "Ошибка: не удалось выделить память\n";return 0;}

//получение интервалов

std::cout << "4.Выбор интервалов.\n"

<< "\t1.Установить интервалы равномерно по диапозону\n"

<< "\t2.Установить интервалы самостоятельно\n";

while(i == 0)

{ std::cin >> i;

switch(i)

{case 1:

len=Xmax-Xmin;

for( i=0;i<NInt;i++){LGrInt[i]=Xmin+len/NInt\*i; }

break;

case 2:

LGrInt[0]=Xmin;

std::cout << "Граница 1: " << Xmin << "\n\n";

for(i=1;i<NInt;i++){

do{

std::cout << "Граница " << i+1 << ": ";

std::cin >> len;

}while(!InsertBorder(LGrInt,i,len,Xmin,Xmax)); }

break;

default:

std::cout << "Ошибка: неверный номер операции, введите номер еще раз: ";

i=0;

break;

}

}

return true;

}

bool InsertBorder(int\* & LGrInt,int cur\_len,int num,int min,int max)

{

if(num < min || num > max){std::cout << "Ошибка: число не в пределах интервала или уже задано\n"; return false; }

int i=0;

int j=0;

while(i<cur\_len)

{ if(LGrInt[i]<num){ i++;

}else if(LGrInt[i]==num)

{ std::cout << "Ошибка: число уже присутствует в массиве\n";

return false;

}else{

j=i;

j=cur\_len;

while(i<j){

LGrInt[j]=LGrInt[j-1];

j--;}

break;

}

}

LGrInt[i]=num;

return true;

}

void GetRandomNum(int\* & Number,int len,int min,int max)

{

std::random\_device rd;

std::mt19937 gen(rd());

std::uniform\_int\_distribution<> distr(min, max);

for(--len;len>=0;len--){

Number[len]=distr(gen);}

}

void InitArray(int\* & Arr,int len)

{

for(--len;len>=0;len--){

Arr[len]=0;}

}

void PrintResult1(int\* Number,int len)

{

int i=0;

std::ofstream fout;

fout.open("result1.txt");

if(!fout.is\_open()){

std::cout << "Error: can't open file " << '\n';

return ;}

std::cout << "Распределение случайных чисел по инервалам еденичной длины:\n";

std::cout << "№\t Лев.гр.\tКл-во\t\n";

fout << "Распределение случайных чисел по инервалам еденичной длины:\n";

fout << "№\t Лев.гр.\tКл-во\t\n";

for(i=0;i<len;i++)

{

std::cout << i << '\t' << i << "\t\t" << Number[i] << '\n';

fout << i << '\t' << i << "\t\t" << Number[i] << '\n';

}

std::cout << "-----------------------------\n";

fout.close();

}

void PrintResult2(int\* LGrInt ,int\* CountNum,int len)

{

int i=0;

std::ofstream fout;

fout.open("result2.txt");

if(!fout.is\_open()){

std::cout << "Error: can't open file " << '\n';

return ;}

std::cout << "Распределение случайных чисел по заданным интервалам:\n";

std::cout << "№\t Лев.гр.\tКл-во\t\n";

fout << "Распределение случайных чисел по инервалам еденичной длины:\n";

fout << "№\t Лев.гр.\tКл-во\t\n";

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

std::cout << i << '\t' << LGrInt[i] << "\t\t" << CountNum[i] << '\n';

fout << i << '\t' << LGrInt[i] << "\t\t" << CountNum[i] << '\n';}

fout.close()}

файл func.asm

.586p

.MODEL FLAT, C

.DATA

offsetcountn dd 0

.CODE

PUBLIC C SORTSTEP1 ;распределение по интервалам единичной длины

SORTSTEP1 PROC C Number:dword, NumRanDat:byte, CountNumUnit1:dword, Xmin:byte

PUSH EDI ;сохранение регистров

PUSH ESI

MOV EAX, DWORD PTR Xmin ;получение значение xmin

MOV EDI, Number ;получение адреса массива случайных чисел

MOV ESI, CountNumUnit1 ;получение адреса массива счетчика чисел

MOV ECX, DWORD PTR NumRanDat ;получение длины массива случайных чисел

cycle:

MOV EBX,[EDI] ; получение числа

SUB EBX,EAX ; вычесть xmin

MOV EDX,[ESI+4\*EBX] ; плучить счетчик этого числа

INC EDX ; прибавить 1

MOV [ESI+4\*EBX],EDX ; занести в обратно массив

ADD EDI,4 ; переход к след числу

LOOP cycle ; повторять пока не прошли весь массив

POP ESI ;восстановление значений регистров

POP EDI

RET

SORTSTEP1 ENDP

PUBLIC C SORTSTEP2 ;распределение по интервалам произвольной длины

SORTSTEP2 PROC C CountNumUnit1:dword, lenUnit1:byte, LGrInt:dword, CountNumN:dword ,NInt:byte, Xmin:byte

PUSH EDI ;сохранение регистров

PUSH ESI

MOV EDI,DWORD PTR lenUnit1 ;получение указателя на последний элемент массива счечика еденичной длины

DEC EDI

SHL EDI,2

ADD EDI,CountNumUnit1

MOV ECX,DWORD PTR lenUnit1 ;поолучение счетчика

MOV EAX, DWORD PTR NInt ;получение указателей по последние элементы массивов

DEC EAX

SHL EAX,2

push edi

MOV ESI, LGrInt

MOV Edi, CountNumN

ADD Edi,EAX

ADD ESI,EAX

MOV offsetcountn,edi

pop edi

; опрерации перед выполнением цикла

SUB EAX,EAX

MOV EAX, dword PTR Xmin

ADD EAX, ECX

DEC EAX

MOV EBX,[ESI]

cycle:

CMP EAX,EBX

JL falseif

PUSH EAX

push esi

mov esi,offsetcountn

MOV EDX,[Esi]

MOV EAX,[EDI]

ADD EDX,EAX

MOV [Esi],EDX

pop esi

POP EAX

JMP endif1

falseif:

SUB ESI,4

SUB offsetcountn,4

MOV EBX,[ESI]

JMP cycle

endif1:

DEC EAX

SUB EDI,4

LOOP cycle ;повторять до тех ПОР ПОКА НЕ ПРОЙДЕМ ВЕСЬ МАССИВ

POP ESI ;восстановление значений регистров

POP EDI

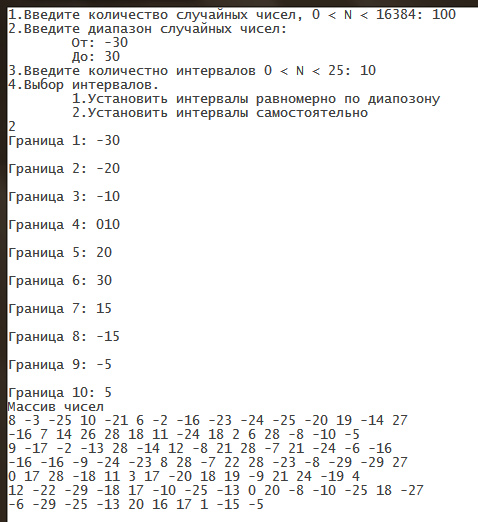
RET

SORTSTEP2 ENDP

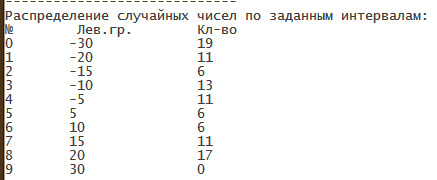
END

# Пример выполнения программы

Исходные данные:



Резульат:



# Вывод

При разработке программы в данной лабораторной работе был изучен и реализован на практике теоретический материал, касающийся организации связи Ассемблера с ЯВУ, на примере программы построения частотного распределения попаданий псевдослучайных целых чисел в заданные интервалы.