**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «Качество и метрология программного обеспечения»**

Тема: **«Расчет метрических характеристик качества разработки программ по метрикам Холстеда»**

|  |  |  |
| --- | --- | --- |
| Студент гр. 6304 |  | Иванов В.С. |
| Преподаватель |  | Кирьянчиков В.А. |

Санкт-Петербург

2020

**Задание**

Для заданного варианта программы обработки данных, представленной на языке Паскаль, разработать вычислительный алгоритм и также варианты программ его реализации на языках программирования Си и Ассемблер. Добиться, чтобы программы на Паскале и Си были работоспособны и давали корректные результаты (это потребуется в дальнейшем при проведении с ними измерительных экспериментов).

Для каждой из разработанных программ (включая исходную программу на Паскале) определить следующие метрические характеристики (по Холстеду):

1. Измеримые характеристики программ:

- число простых(отдельных)операторов, в данной реализации;

- число простых (отдельных) операндов, в данной реализации;

- общее число всех операторов в данной реализации;

- общее число всех операндов в данной реализации;

- число вхождений j-го оператора в тексте программы;

- число вхождений j-го операнда в тексте программы;

- словарь программы;

- длину программы.

2. Расчетные характеристики программы:

- длину программы;

- реальный и потенциальный объемы программы;

- уровень программы;

- интеллектуальное содержание программы;

- работу программиста;

- время программирования;

- уровень используемого языка программирования;

- ожидаемое число ошибок в программе.

Для характеристик длина программы, уровень программы, время программирования следует рассчитать как саму характеристику, так и ее оценку.

**Ход работы**

1. Определение метрических характеристик для программы на Pascal.

Код программы представлен в приложении А.

Ручной расчёт измеримых характеристик представлен в таблице 1.

Таблица 1 – Ручной расчёт измеримых характеристик (Pascal)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Оператор | Количество | № | Операнд | Количество |
| 1 | ; | 27 | 1 | x | 3 |
| 2 | := | 13 | 2 | y | 2 |
| 3 | () или begin end | 16 | 3 | i | 10 |
| 4 | [] | 12 | 4 | j | 11 |
| 5 | + | 4 | 5 | a | 12 |
| 6 | - | 2 | 6 | n | 5 |
| 7 | > | 2 | 7 | p | 3 |
| 8 | for to do | 4 | 8 | q | 3 |
| 9 | If then | 2 | 9 | hold | 6 |
| 10 | repeat until | 1 | 10 | no\_change | 4 |
| 11 | sort1 | 1 | 11 | 1 | 9 |
| 12 | sort2 | 1 | 12 | 1000 | 3 |
| 13 | swap | 1 | 13 | 999 | 1 |
| 14 | randomize | 1 | 14 | true | 1 |
| 15 | random | 1 | 15 | false | 1 |
| Всего | | 88 | Всего | | 74 |

Программный расчёт измеримых характеристик представлен в таблицу 2. Файл с результатами программных расчётов представлен в приложении Б.

Таблица 2 – Программный расчёт измеримых характеристик (Pascal)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Оператор | Количество | № | Операнд | Количество |
| 1 | () | 10 | 1 | 1 | 10 |
| 2 | + | 4 | 2 | 1000 | 4 |
| 3 | - | 2 | 3 | 999 | 1 |
| 4 | ; | 39 | 4 | a | 12 |
| 5 | = | 10 | 5 | bubble\_sort | 1 |
| 6 | > | 2 | 6 | false | 1 |
| 7 | [] | 13 | 7 | hold | 6 |
| 8 | boolean | 1 | 8 | i | 9 |
| 9 | for | 4 | 9 | j | 9 |
| 10 | if | 2 | 10 | n | 6 |
| 11 | Integer | 5 | 11 | no\_change | 4 |
| 12 | procedure | 3 | 12 | p | 3 |
| 13 | program | 1 | 13 | q | 3 |
| 14 | random | 1 | 14 | true | 1 |
| 15 | randomize | 1 | 15 | x | 4 |
| 16 | real | 6 | 16 | y | 3 |
| 17 | repeat | 1 | Всего | | 77 |
| 18 | sort1 | 2 |
| 19 | sort2 | 2 |
| 20 | swap | 2 |
| Всего | | 111 |

Определение расчетных характеристик представлено в таблице 3.

Таблица 3 – Расчёт расчетных характеристик (Pascal)

|  |  |  |
| --- | --- | --- |
| Характеристика | Ручной расчёт | Программный расчёт |
| Число простых операторов n1 | 15 | 20 |
| Число простых операндов n2 | 15 | 16 |
| Общее число всех операторов N1 | 88 | 111 |
| Общее число всех операндов N2 | 74 | 77 |
| Словарь n | 30 | 36 |
| Длина Nопыт | 162 | 188 |
| Теоретическая длина Nтеор | 190 | 150 |
| Объём V | 794.916 | 971.946 |
| Потенциальный объём V\* | 19.65 | 19.65 |
| Уровень программы L | 0.026 | 0.02 |
| Оценка уровня программы L~ | 0.027 | 0.021 |
| Интеллектуальное содержание I | 21.46 | 20.19 |
| Работа программирования E | 30574 | 48071 |
| Оценка времени программирования T^ | 3057.4 | 2079.4 |
| Время программирования T | 2941.2 | 2670.65 |
| Уровень языка λ | 0.486 | 0.397 |
| Ожидаемое число ошибок в программе B | 1.99 | 0.44 |

1. Определение метрических характеристик для программы на Си.

Код программы представлен в приложении В.

Ручной расчёт измеримых характеристик представлен в таблице 4.

Таблица 4 – Ручной расчёт измеримых характеристик (Си)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Оператор | Количество | № | Операнд | Количество |
| 1 | ; | 25 | 1 | x | 14 |
| 2 | = | 15 | 2 | y | 2 |
| 3 | () или {} | 27 | 3 | i | 12 |
| 4 | [] | 13 | 4 | n | 5 |
| 5 | for | 4 | 5 | a | 3 |
| 6 | if | 2 | 6 | hold | 5 |
| 7 | > | 2 | 7 | no\_change | 4 |
| 8 | < | 4 | 8 | j | 13 |
| 9 | + | 4 | 9 | 0 | 6 |
| 10 | ++ | 4 | 10 | 1 | 7 |
| 11 | - | 2 | 11 | 999 | 1 |
| 12 | % | 1 | 12 | 1000 | 3 |
| 13 | \* | 7 | 13 | NULL | 1 |
| 14 | & | 2 | Всего | | 76 |
| 15 | return | 1 |
| 16 | sort1 | 1 |
| 17 | sort2 | 1 |
| 18 | swap | 1 |
| 19 | srand | 1 |
| 20 | time | 1 |
| 21 | rand | 1 |
| 22 | ! | 1 |
| Всего | | 120 |

Программный расчёт измеримых характеристик представлен в таблицу 5. Файл с результатами программных расчётов представлен в приложении Г.

Таблица 5 – Программный расчёт измеримых характеристик (Си)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Оператор | Количество | № | Операнд | Количество |
| 1 | ! | 1 | 1 | 0 | 6 |
| 2 | % | 1 | 2 | 1 | 7 |
| 3 | () | 13 | 3 | 1000 | 5 |
| 4 | + | 4 | 4 | 999 | 1 |
| 5 | ++ | 4 | 5 | NULL | 1 |
| 6 | , | 6 | 6 | a | 3 |
| 7 | - | 2 | 7 | b | 3 |
| 8 | ; | 35 | 8 | hold | 5 |
| 9 | < | 4 | 9 | i | 13 |
| 10 | = | 15 | 10 | j | 13 |
| 11 | > | 2 | 11 | n | 5 |
| 12 | [] | 13 | 12 | no\_change | 4 |
| 13 | \_& | 2 | 13 | x | 16 |
| 14 | \_\* | 4 | 14 | y | 3 |
| 15 | \_[] | 2 | Всего | | 85 |
| 16 | \_\_\* | 4 |
| 17 | float | 8 |
| 18 | for | 4 |
| 19 | if | 2 |
| 20 | int | 8 |
| 21 | main | 1 |
| 22 | rand | 1 |
| 23 | return | 1 |
| 24 | sort1 | 2 |
| 25 | sort2 | 2 |
| 26 | srand | 1 |
| 27 | swap | 2 |
| 28 | time | 1 |
| 29 | void | 3 |
| 30 | while | 1 |
| Всего | | 149 |

Определение расчетных характеристик представлено в таблице 6.

Таблица 6 – Расчёт расчетных характеристик (Си)

|  |  |  |
| --- | --- | --- |
| Характеристика | Ручной расчёт | Программный расчёт |
| Число простых операторов n1 | 22 | 30 |
| Число простых операндов n2 | 13 | 14 |
| Общее число всех операторов N1 | 120 | 149 |
| Общее число всех операндов N2 | 76 | 85 |
| Словарь n | 35 | 44 |
| Длина Nопыт | 196 | 234 |
| Теоретическая длина Nтеор | 241.7 | 200.51 |
| Объём V | 1044.3 | 1277.5 |
| Потенциальный объём V\* | 19.65 | 19.65 |
| Уровень программы L | 0.019 | 0.015 |
| Оценка уровня программы L~ | 0.015 | 0.011 |
| Интеллектуальное содержание I | 15.66 | 14.03 |
| Работа программирования E | 54963 | 83048 |
| Оценка времени программирования T^ | 5496.3 | 5538.5 |
| Время программирования T | 7021 | 4613 |
| Уровень языка λ | 0.37 | 0.302 |
| Ожидаемое число ошибок в программе B | 2.61 | 0.63 |

1. Определение метрических характеристик для программы на Си.

Код программы представлен в приложении Д.

Ручной расчёт измеримых характеристик представлен в таблице 7.

Таблица 7 – Ручной расчёт измеримых характеристик (Ассемблер)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Оператор | Количество | № | Операнд | Количество |
| 1 | pushq | 4 | 1 | %rbp | 74 |
| 2 | movq | 29 | 2 | %rsp | 6 |
| 3 | movl | 38 | 3 | %rdi | 6 |
| 4 | jmp .L2 | 1 | 4 | -24(%rbp) | 15 |
| 5 | addl | 6 | 5 | %esi | 4 |
| 6 | jmp .L3 | 1 | 6 | -28(%rbp) | 5 |
| 7 | cltq | 13 | 7 | $0 | 8 |
| 8 | leaq | 12 | 8 | -12(%rbp) | 7 |
| 9 | addq | 12 | 9 | %eax | 43 |
| 10 | movss | 19 | 10 | $1 | 13 |
| 11 | ucomiss | 2 | 11 | -8(%rbp) | 12 |
| 12 | jbe .L4 | 1 | 12 | 0(,%rax,4) | 10 |
| 13 | cmpl | 5 | 13 | %rdx | 19 |
| 14 | jl .L6 | 1 | 14 | %rax | 57 |
| 15 | subl | 4 | 15 | %xmm0 | 20 |
| 16 | jl .L7 | 1 | 16 | %xmm1 | 4 |
| 17 | nop | 3 | 17 | -4(%rbp) | 11 |
| 18 | popq | 2 | 18 | %rcx | 2 |
| 19 | ret | 4 | 19 | %rsi | 4 |
| 20 | jmp .L11 | 1 | 20 | -32(%rbp) | 3 |
| 21 | jmp .L12 | 1 | 21 | $32 | 1 |
| 22 | jbe .L13 | 1 | 22 | $8032 | 1 |
| 23 | call swap | 1 | 23 | %fs:40 | 2 |
| 24 | jl .L15 | 1 | 24 | -8020(%rbp) | 6 |
| 25 | je .L16 | 1 | 25 | %ecx | 5 |
| 26 | leave | 2 | 26 | %edx | 5 |
| 27 | subq | 2 | 27 | $9 | 1 |
| 28 | xorl | 1 | 28 | $31 | 1 |
| 29 | call time@PLT | 1 | 29 | $999 | 2 |
| 30 | call srand@PLT | 1 | 30 | -8016(%rbp,%rax,4) | 2 |
| 31 | call rand@PLT | 1 | 31 | -4016(%rbp,%rax,4) | 1 |
| 32 | imull | 2 | 32 | -8016(%rbp) | 1 |
| 33 | leal | 1 | 33 | $1000 | 2 |
| 34 | sarl | 2 | 34 | -4016(%rbp) | 1 |
| 35 | cvtsi2ss | 1 | Всего | | 354 |
| 36 | jle .L20 | 1 |
| 37 | call sort1 | 1 |
| 38 | call sort2 | 1 |
| 39 | je .L22 | 1 |
| Всего | | 182 |

Определение расчетных характеристик представлено в таблице 8.

Таблица 8 – Расчёт расчетных характеристик (Ассемблер)

|  |  |
| --- | --- |
| Характеристика | Ручной расчёт |
| Число простых операторов n1 | 39 |
| Число простых операндов n2 | 34 |
| Общее число всех операторов N1 | 182 |
| Общее число всех операндов N2 | 354 |
| Словарь n | 73 |
| Длина Nопыт | 536 |
| Теоретическая длина Nтеор | 580.7 |
| Объём V | 3317.7 |
| Потенциальный объём V\* | 19.65 |
| Уровень программы L | 0.006 |
| Оценка уровня программы L~ | 0.005 |
| Интеллектуальное содержание I | 16.59 |
| Работа программирования E | 552950 |
| Оценка времени программирования T^ | 55295 |
| Время программирования T | 67360 |
| Уровень языка λ | 0.12 |
| Ожидаемое число ошибок в программе B | 8.3 |

1. Сравнение результатов определения метрических характеристик.

Таблица 9 – Сводная таблица расчетов на трех языках

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Характеристика | Ручной расчёт Pascal | Програм-мный расчёт Pascal | Ручной расчёт Си | Програм-мный расчёт Си | Ручной расчёт Ассемблер |
| Число простых операторов n1 | 15 | 20 | 22 | 30 | 39 |
| Число простых операндов n2 | 15 | 16 | 13 | 14 | 34 |
| Общее число всех операторов N1 | 88 | 111 | 120 | 149 | 182 |
| Общее число всех операндов N2 | 74 | 77 | 76 | 85 | 354 |
| Словарь n | 30 | 36 | 35 | 44 | 73 |
| Длина Nопыт | 162 | 188 | 196 | 234 | 536 |
| Теоретическая длина Nтеор | 190 | 150 | 241.7 | 200.51 | 580.7 |
| Объём V | 794.916 | 971.946 | 1044.3 | 1277.5 | 3317.7 |
| Потенциальный объём V\* | 19.65 | 19.65 | 19.65 | 19.65 | 19.65 |
| Уровень программы | 0.026 | 0.02 | 0.019 | 0.015 | 0.006 |
| Оценка уровня программы L~ | 0.027 | 0.021 | 0.015 | 0.011 | 0.005 |
| Интеллектуальное содержание I | 21.46 | 20.19 | 15.66 | 14.03 | 16.59 |
| Работа программирования E | 30574 | 48071 | 54963 | 83048 | 552950 |
| Оценка времени программирования T^ | 3057.4 | 2079.4 | 5496.3 | 5538.5 | 55295 |
| Время программирования T | 2941.2 | 2670.65 | 7021 | 4613 | 67360 |
| Уровень языка λ | 0.486 | 0.397 | 0.37 | 0.302 | 0.12 |
| Ожидаемое число ошибок в программе B | 1.99 | 0.44 | 2.61 | 0.63 | 8.3 |

В результате сравнения видно, что уровень программы самый низкий у программы на Ассемблере, а самый высокий у программы на Pascal. Наибольшие показатели времени программирования, работы программирования и ожидаемого числа ошибок, наоборот, соответствуют Ассемблеру, а наименьший – Pascal.

**Выводы**

В результате выполнения данной лабораторной работы была изучена система метрик Холстеда. Было проведено сравнение программ, реализующих алгоритмы сортировки пузырьком, на языках Pascal, Си и Ассемблер.

**ПРИЛОЖЕНИЕ А**

**Код программы на Pascal.**

program bubble\_sort;

var x,y: array[1..1000] of real; i,n: integer;

procedure sort1(var a: array of real; n: integer);

var i,j: integer; hold: real;

begin

for i:=1 to n-1 do

for j:=i+1 to n do

begin

if a[i]>a[j] then

begin

hold:=a[i];

a[i]:=a[j];

a[j]:=hold;

end

end

end;

procedure sort2(var a: array of real; n: integer);

var no\_change: boolean; j: integer;

procedure swap(var p, q: real);

var hold: real;

begin

hold:=p;

p:=q;

q:=hold;

end;

begin

repeat

no\_change:=true;

for j:=1 to n-1 do

begin

if a[j]>a[j+1] then

begin

swap(a[j],a[j+1]);

no\_change:=false;

end

end

until no\_change;

end;

begin

randomize;

for i:= 1 to 1000 do

begin

x[i] := random(999) + 1;

y[i] := x[i];

end;

sort1(x, 1000);

sort2(y, 1000);

end.

**ПРИЛОЖЕНИЕ Б**

**Результаты parser\_pas.exe**

Statistics for module Z:\pasout.lxm

=====================================

The number of different operators : 20

The number of different operands : 16

The total number of operators : 111

The total number of operands : 77

Dictionary ( D) : 36

Length ( N) : 188

Length estimation ( ^N) : 150.439

Volume ( V) : 971.946

Potential volume ( \*V) : 19.6515

Limit volume (\*\*V) : 38.2071

Programming level ( L) : 0.0202187

Programming level estimation ( ^L) : 0.0207792

Intellect ( I) : 20.1963

Time of programming ( T) : 2670.65

Time estimation ( ^T) : 2079.42

Programming language level (lambda) : 0.397328

Work on programming ( E) : 48071.6

Error ( B) : 0.440695

Error estimation ( ^B) : 0.323982

Table:

====================================

Operators:

| 1 | 10 | ()

| 2 | 4 | +

| 3 | 2 | -

| 4 | 39 | ;

| 5 | 10 | =

| 6 | 2 | >

| 7 | 13 | []

| 8 | 1 | boolean

| 9 | 4 | for

| 10 | 2 | if

| 11 | 5 | integer

| 12 | 3 | procedure

| 13 | 1 | program

| 14 | 1 | random

| 15 | 1 | randomize

| 16 | 6 | real

| 17 | 1 | repeat

| 18 | 2 | sort1

| 19 | 2 | sort2

| 20 | 2 | swap

Operands:

| 1 | 10 | 1

| 2 | 4 | 1000

| 3 | 1 | 999

| 4 | 12 | a

| 5 | 1 | bubble\_sort

| 6 | 1 | false

| 7 | 6 | hold

| 8 | 9 | i

| 9 | 9 | j

| 10 | 6 | n

| 11 | 4 | no\_change

| 12 | 3 | p

| 13 | 3 | q

| 14 | 1 | true

| 15 | 4 | x

| 16 | 3 | y

Summary:

=====================================

The number of different operators : 20

The number of different operands : 16

The total number of operators : 111

The total number of operands : 77

Dictionary ( D) : 36

Length ( N) : 188

Length estimation ( ^N) : 150.439

Volume ( V) : 971.946

Potential volume ( \*V) : 19.6515

Limit volume (\*\*V) : 38.2071

Programming level ( L) : 0.0202187

Programming level estimation ( ^L) : 0.0207792

Intellect ( I) : 20.1963

Time of programming ( T) : 2670.65

Time estimation ( ^T) : 2079.42

Programming language level (lambda) : 0.397328

Work on programming ( E) : 48071.6

Error ( B) : 0.440695

Error estimation ( ^B) : 0.323982

**ПРИЛОЖЕНИЕ В**

**Код программы на Си**

#include <math.h>

#include <stdlib.h>

#include <time.h>

#include <stdio.h>

void sort1(float\* x, int n){

float hold;

for (int i = 0; i < n - 1; i++) {

for (int j = i + 1; j < n; j++) {

if (x[i] > x[j]) {

hold = x[i];

x[i] = x[j];

x[j] = hold;

}

}

}

}

void swap(float \*a, float \*b) {

float hold = (\*a);

\*a = (\*b);

\*b = hold;

}

void sort2(float \*x, int n){

int no\_change = 0;

while(!no\_change) {

no\_change = 1;

for (int j=0; j < n-1; j++) {

if (x[j] > x[j+1]) {

swap(&x[j], &x[j+1]);

no\_change = 0;

}

}

}

}

int main(){

float x[1000];

float y[1000];

srand(time(NULL));

for (int i=0; i <1000; i++) {

x[i] = 1 + rand() % 999;

y[i] = x[i];

}

sort1(x,1000);

sort2(y,1000);

return 0;

}

**ПРИЛОЖЕНИЕ Г**

**Результаты parser\_c.exe**

Statistics for module Z:\output.lxm

=====================================

The number of different operators : 30

The number of different operands : 14

The total number of operators : 149

The total number of operands : 85

Dictionary ( D) : 44

Length ( N) : 234

Length estimation ( ^N) : 200.51

Volume ( V) : 1277.51

Potential volume ( \*V) : 19.6515

Limit volume (\*\*V) : 38.2071

Programming level ( L) : 0.0153827

Programming level estimation ( ^L) : 0.0109804

Intellect ( I) : 14.0275

Time of programming ( T) : 4613.8

Time estimation ( ^T) : 5538.5

Programming language level (lambda) : 0.302293

Work on programming ( E) : 83048.4

Error ( B) : 0.634502

Error estimation ( ^B) : 0.425836

Table:

====================================

Operators:

| 1 | 1 | !

| 2 | 1 | %

| 3 | 13 | ()

| 4 | 4 | +

| 5 | 4 | ++

| 6 | 6 | ,

| 7 | 2 | -

| 8 | 35 | ;

| 9 | 4 | <

| 10 | 15 | =

| 11 | 2 | >

| 12 | 13 | []

| 13 | 2 | \_&

| 14 | 4 | \_\*

| 15 | 2 | \_[]

| 16 | 4 | \_\_\*

| 17 | 8 | float

| 18 | 4 | for

| 19 | 2 | if

| 20 | 8 | int

| 21 | 1 | main

| 22 | 1 | rand

| 23 | 1 | return

| 24 | 2 | sort1

| 25 | 2 | sort2

| 26 | 1 | srand

| 27 | 2 | swap

| 28 | 1 | time

| 29 | 3 | void

| 30 | 1 | while

Operands:

| 1 | 6 | 0

| 2 | 7 | 1

| 3 | 5 | 1000

| 4 | 1 | 999

| 5 | 1 | NULL

| 6 | 3 | a

| 7 | 3 | b

| 8 | 5 | hold

| 9 | 13 | i

| 10 | 13 | j

| 11 | 5 | n

| 12 | 4 | no\_change

| 13 | 16 | x

| 14 | 3 | y

Summary:

=====================================

The number of different operators : 30

The number of different operands : 14

The total number of operators : 149

The total number of operands : 85

Dictionary ( D) : 44

Length ( N) : 234

Length estimation ( ^N) : 200.51

Volume ( V) : 1277.51

Potential volume ( \*V) : 19.6515

Limit volume (\*\*V) : 38.2071

Programming level ( L) : 0.0153827

Programming level estimation ( ^L) : 0.0109804

Intellect ( I) : 14.0275

Time of programming ( T) : 4613.8

Time estimation ( ^T) : 5538.5

Programming language level (lambda) : 0.302293

Work on programming ( E) : 83048.4

Error ( B) : 0.634502

Error estimation ( ^B) : 0.425836

**ПРИЛОЖЕНИЕ Д**

**Код программы на Ассемблер**

sort1:

.LFB5:

.cfi\_startproc

pushq %rbp

.cfi\_def\_cfa\_offset 16

.cfi\_offset 6, -16

movq %rsp, %rbp

.cfi\_def\_cfa\_register 6

movq %rdi, -24(%rbp)

movl %esi, -28(%rbp)

movl $0, -12(%rbp)

jmp .L2

.L7:

movl -12(%rbp), %eax

addl $1, %eax

movl %eax, -8(%rbp)

jmp .L3

.L6:

movl -12(%rbp), %eax

cltq

leaq 0(,%rax,4), %rdx

movq -24(%rbp), %rax

addq %rdx, %rax

movss (%rax), %xmm0

movl -8(%rbp), %eax

cltq

leaq 0(,%rax,4), %rdx

movq -24(%rbp), %rax

addq %rdx, %rax

movss (%rax), %xmm1

ucomiss %xmm1, %xmm0

jbe .L4

movl -12(%rbp), %eax

cltq

leaq 0(,%rax,4), %rdx

movq -24(%rbp), %rax

addq %rdx, %rax

movss (%rax), %xmm0

movss %xmm0, -4(%rbp)

movl -8(%rbp), %eax

cltq

leaq 0(,%rax,4), %rdx

movq -24(%rbp), %rax

addq %rax, %rdx

movl -12(%rbp), %eax

cltq

leaq 0(,%rax,4), %rcx

movq -24(%rbp), %rax

addq %rcx, %rax

movss (%rdx), %xmm0

movss %xmm0, (%rax)

movl -8(%rbp), %eax

cltq

leaq 0(,%rax,4), %rdx

movq -24(%rbp), %rax

addq %rdx, %rax

movss -4(%rbp), %xmm0

movss %xmm0, (%rax)

.L4:

addl $1, -8(%rbp)

.L3:

movl -8(%rbp), %eax

cmpl -28(%rbp), %eax

jl .L6

addl $1, -12(%rbp)

.L2:

movl -28(%rbp), %eax

subl $1, %eax

cmpl %eax, -12(%rbp)

jl .L7

nop

popq %rbp

.cfi\_def\_cfa 7, 8

ret

.cfi\_endproc

.LFE5:

.size sort1, .-sort1

.globl swap

.type swap, @function

swap:

.LFB6:

.cfi\_startproc

pushq %rbp

.cfi\_def\_cfa\_offset 16

.cfi\_offset 6, -16

movq %rsp, %rbp

.cfi\_def\_cfa\_register 6

movq %rdi, -24(%rbp)

movq %rsi, -32(%rbp)

movq -24(%rbp), %rax

movss (%rax), %xmm0

movss %xmm0, -4(%rbp)

movq -32(%rbp), %rax

movss (%rax), %xmm0

movq -24(%rbp), %rax

movss %xmm0, (%rax)

movq -32(%rbp), %rax

movss -4(%rbp), %xmm0

movss %xmm0, (%rax)

nop

popq %rbp

.cfi\_def\_cfa 7, 8

ret

.cfi\_endproc

.LFE6:

.size swap, .-swap

.globl sort2

.type sort2, @function

sort2:

.LFB7:

.cfi\_startproc

pushq %rbp

.cfi\_def\_cfa\_offset 16

.cfi\_offset 6, -16

movq %rsp, %rbp

.cfi\_def\_cfa\_register 6

subq $32, %rsp

movq %rdi, -24(%rbp)

movl %esi, -28(%rbp)

movl $0, -8(%rbp)

jmp .L11

.L16:

movl $1, -8(%rbp)

movl $0, -4(%rbp)

jmp .L12

.L15:

movl -4(%rbp), %eax

cltq

leaq 0(,%rax,4), %rdx

movq -24(%rbp), %rax

addq %rdx, %rax

movss (%rax), %xmm0

movl -4(%rbp), %eax

cltq

addq $1, %rax

leaq 0(,%rax,4), %rdx

movq -24(%rbp), %rax

addq %rdx, %rax

movss (%rax), %xmm1

ucomiss %xmm1, %xmm0

jbe .L13

movl -4(%rbp), %eax

cltq

addq $1, %rax

leaq 0(,%rax,4), %rdx

movq -24(%rbp), %rax

addq %rax, %rdx

movl -4(%rbp), %eax

cltq

leaq 0(,%rax,4), %rcx

movq -24(%rbp), %rax

addq %rcx, %rax

movq %rdx, %rsi

movq %rax, %rdi

call swap

movl $0, -8(%rbp)

.L13:

addl $1, -4(%rbp)

.L12:

movl -28(%rbp), %eax

subl $1, %eax

cmpl %eax, -4(%rbp)

jl .L15

.L11:

cmpl $0, -8(%rbp)

je .L16

nop

leave

.cfi\_def\_cfa 7, 8

ret

.cfi\_endproc

.LFE7:

.size sort2, .-sort2

.globl main

.type main, @function

main:

.LFB8:

.cfi\_startproc

pushq %rbp

.cfi\_def\_cfa\_offset 16

.cfi\_offset 6, -16

movq %rsp, %rbp

.cfi\_def\_cfa\_register 6

subq $8032, %rsp

movq %fs:40, %rax

movq %rax, -8(%rbp)

xorl %eax, %eax

movl $0, %edi

call time@PLT

movl %eax, %edi

call srand@PLT

movl $0, -8020(%rbp)

jmp .L19

.L20:

call rand@PLT

movl %eax, %ecx

movl $-2093742815, %edx

movl %ecx, %eax

imull %edx

leal (%rdx,%rcx), %eax

sarl $9, %eax

movl %eax, %edx

movl %ecx, %eax

sarl $31, %eax

subl %eax, %edx

movl %edx, %eax

imull $999, %eax, %eax

subl %eax, %ecx

movl %ecx, %eax

addl $1, %eax

cvtsi2ss %eax, %xmm0

movl -8020(%rbp), %eax

cltq

movss %xmm0, -8016(%rbp,%rax,4)

movl -8020(%rbp), %eax

cltq

movss -8016(%rbp,%rax,4), %xmm0

movl -8020(%rbp), %eax

cltq

movss %xmm0, -4016(%rbp,%rax,4)

addl $1, -8020(%rbp)

.L19:

cmpl $999, -8020(%rbp)

jle .L20

leaq -8016(%rbp), %rax

movl $1000, %esi

movq %rax, %rdi

call sort1

leaq -4016(%rbp), %rax

movl $1000, %esi

movq %rax, %rdi

call sort2

movl $0, %eax

movq -8(%rbp), %rsi

xorq %fs:40, %rsi

je .L22

call \_\_stack\_chk\_fail@PLT

.L22:

leave

.cfi\_def\_cfa 7, 8

ret

.cfi\_endproc