

**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 разработки программ по**  
**метрикам Холстеда**

Студент гр. 7304

\_\_\_\_\_

Субботин А.С.

Преподаватель

\_\_\_\_\_

Ефремов М.А.

Санкт-Петербург

2021

## Цель работы

Изучение и сравнение метрик Холстеда для программ на C, Pascal и ассемблере.

## Исходные данные

Вариант 17. Вычисление функции ошибок распределения Гаусса

## Ход работы

1. Из кода предоставленной программы на языке Паскаль исключены операции ввода и вывода данных, результат представлен в Приложении А.

1.1. Произведен ручной расчет измеримых характеристик для программы на языке Паскаль. Результат представлен в Таблице 1.

№	Оператор	Количество	№	Операнд	Количество
1	program	1	1	0.0	4
2	erf	1	2	0.1693122	1
3	erfc	1	3	0.7619048	1
4	exp	2	4	0.6666667	2
5	begin...end	8	5	1	14
6	repeat until	1	6	1.0	5
7	if then else	3	7	1.5	1
8	+	23	8	1.7724538	2
9	*	31	9	10	1
10	/	14	10	11	1
11	:=	18	11	12	3
12	=	1	12	2	2
13	()	25	13	2.0	3
14	-	4	14	3	3
15	<	2	15	3.078403E-3	1
	<b>Всего</b>	<b>135</b>	16	4	1
			17	4.736005E-4	1
			18	5	1

			19	6	1
			20	6.314673E-5	1
			21	6.476214E-9	1
			22	7	1
			23	7.429027E-6	1
			24	7.447646E-8	1
			25	7.820028E-7	1
			26	8	1
			27	9	1
			28	x	17
			29	er	22
			30	ec	6
			31	done	4
			32	sqrtpi	4
			33	t2	2
			34	t3	2
			35	t4	2
			36	t5	2
			37	t6	2
			38	t7	2
			39	t8	2
			40	t9	2
			41	t10	2
			42	t11	2
			43	t12	2
			44	x2	18
			45	sum	8
			46	v	14
			47	true	1
			48	false	1
				<b>Всего</b>	<b>176</b>

Таблица 1 – ручной расчет измеримых характеристик в программе на языке Паскаль

1.2.Произведен программный расчет измеримых характеристик для программы на языке Паскаль. Результат представлен в Таблице 2.

№	Оператор	Количество	№	Операнд	Количество
1	repeat	1	1	0.0	3
2	<	2	2	0.1693122	1
3	()	37	3	0.7619048	1
4	*	31	4	0.6666667	2
5	+	23	5	1	14
6	-	4	6	1.0	5
7	/	14	7	1.5	1
8	=	32	8	1.7724538	2
9	const	2	9	10	1
10	erf	2	10	11	1
11	erfc	2	11	12	3
12	exp	2	12	2	2
13	program	1	13	2.0	3
14	real	2	14	3	3
	<b>Всего</b>	<b>158</b>	15	3.078403E-3	1
			16	4	1
			17	4.736005E-4	1
			18	5	1
			19	6	1
			20	6.314673E-5	1
			21	6.476214E-9	1
			22	7	1
			23	7.429027E-6	1
			24	7.447646E-8	1
			25	7.820028E-7	1
			26	8	1
			27	9	1
			28	done	4
			29	ec	5
			30	er	5

			31	erf	1
			32	erfc	1
			33	erfd4	1
			34	false	1
			35	sqrtpi	4
			36	sum	8
			37	t2	2
			38	t3	2
			39	t4	2
			40	t5	2
			41	t6	2
			42	t7	2
			43	t8	2
			44	t9	2
			45	t10	2
			46	t11	2
			47	t12	2
			48	true	1
			49	v	14
			50	x	17
			51	x2	18
				<b>Всего</b>	<b>152</b>

Таблица 2 – программный расчет измеримых характеристик в программе  
на языке Паскаль

1.3. Расчетные характеристики вычислены вручную и с помощью программы. Результаты представлены в Таблице 3.

Характеристика	Ручной расчет	Программный расчет
Число простых операторов $n_1$	15	15
Число простых операндов $n_2$	49	51
Общее число всех операторов $N_1$	135	158
Общее число всех операндов $N_2$	176	152
Словарь $n = n_1 + n_2$	64	66
Длина $N = N_1 + N_2$	311	310
Потенциальная теоретич. длина $N_{\text{теор}} = \sum n_i \log_2(n_i)$	333,72	347.90
Объем $V = N \log_2(n)$	1866	1873.76
Потенциальный объем $V^* = (n_2^* + 2) \log_2(n_2^* + 2)$ , $n_2^* = 3$	11,61	11.61
Уровень $L = V^*/V$	0,006	0.006
Интеллектуальное содержание $I = 2/n_1 * n_2/N_2 * V$	69,27	83.83
Работа по программированию $E = V/L$	299919,36	302420
Время программирования $T = E/S$ , $S = 10$	29991,94	16801.10
Уровень языка $h = LV^*$	0,07	0.07
Количество ошибок $B = V/1000$	2	2

Таблица 3 – сводная таблица с расчетными характеристиками для программы на языке Паскаль

2. На основе программы на языке Паскаль была написана аналогичная программа на языке Си. Для корректной отработки программы расчета измерительных и расчетных характеристик пришлось избавиться от глобальных переменных. Код программы представлен в Приложении Б.

2.1. Произведен ручной расчет измеримых характеристик для программы на языке Си. Результат представлен в Таблице 4.

№	Оператор	Количество	№	Операнд	Количество
1	erf	4	1	x	17
2	erfc	2	2	er	5
3	Return	3	3	ec	5
4	If else	3	4	done	4
5	Do while	1	5	sqrtpi	4
6	+	23	6	t2	2
7	-	4	7	t3	2
8	*	31	8	t4	2
9	/	14	9	t5	2
10	Exp	2	10	t6	2
11	()	27	11	t7	2
12	Main	1	12	t8	2
13	<	1	13	t9	2
14	{ }	8	14	t10	2
	<b>Всего</b>	<b>124</b>	15	t11	2
			16	t12	2
			17	x2	18
			18	sum	8
			19	v	14
			20	false	2
			21	true	1
			22	0	4
			23	0.1693122	1
			24	0.7619048	1
			25	0.6666667	2
			26	1	17
			27	1.0	1
			28	1.5	1
			29	1.7724538	2
			30	10	1

			31	11	1
			32	12	1
			33	2	2
			34	2.0	3
			35	3	3
			36	3.078403E-3	1
			37	4	1
			38	4.736005E-4	1
			39	5	1
			40	6	1
			41	6.314673E-5	1
			42	6.476214E-9	1
			43	7	1
			44	7.429027E-6	1
			45	7.447646E-8	1
			46	7.820028E-7	1
			47	8	1
			48	9	1
				<b>Всего</b>	<b>192</b>

Таблица 4 – ручной расчет измеримых характеристик в программе на языке Си



2.2.Произведен программный расчет измеримых характеристик для программы на языке Си. Результат представлен в Таблице 5.

№	Оператор	Количество	№	Операнд	Количество
1	erf	1	1	x	9
2	erfc	1	2	er	1
3	Return	2	3	ec	1
4	,	5	4	done	4
5	+	23	5	sqrtpi	4
6	-	1	6	t2	2
7	*	31	7	t3	2
8	/	14	8	t4	2
9	Exp	2	9	t5	2
10	()	29	10	t6	2
11	=	19	11	t7	2
	<b>Всего</b>	<b>128</b>	12	t8	2
			13	t9	2
			14	t10	2
			15	t11	2
			16	t12	2
			17	x2	18
			18	sum	8
			19	v	14
			20	0.1693122	1
			21	0.7619048	1
			22	0.6666667	2
			23	1	14
			24	1.0	1
			25		
			26	1.7724538	2
			27	10	1
			28	11	1
			29	12	1
			30	2	2

			31	2.0	1
			32	3	1
			33	3.078403E-3	1
			34	4	1
			35	4.736005E-4	1
			36	5	1
			37	6	1
			38	6.314673E-5	1
			39	6.476214E-9	1
			40	7	1
			41	7.429027E-6	1
			42	7.447646E-8	1
			43	7.820028E-7	1
			44	8	1
			45	9	1
				<b>Всего</b>	<b>118</b>

Таблица 5 – программный расчет измеримых характеристик в программе  
на языке Си

2.3. Расчетные характеристики вычислены вручную и с помощью программы. Результаты представлены в Таблице 6.

Характеристика	Ручной расчет	Программный расчет
Число простых операторов $n_1$	14	11
Число простых операндов $n_2$	48	43
Общее число всех операторов $N_1$	124	128
Общее число всех операндов $N_2$	192	118
Словарь $n = n_1 + n_2$	62	54
Длина $N = N_1 + N_2$	316	246
Потенциальная теоретич. длина $N_{\text{теор}} = \sum n_i \log_2(n_i)$	321,38	271,38
Объем $V = N \log_2(n)$	1881,53	1415,70
Потенциальный объем $V^* = (n_2^* + 2) \log_2(n_2^* + 2)$ , $n_2^* = 3$	11,61	11,61
Уровень $L = V^*/V$	0,006	0,008
Интеллектуальное содержание $I = 2/n_1 * n_2/N_2 * V$	67,19	93,80
Работа по программированию $E = V/L$	304931,08	172634
Время программирования $T = E/S$ , $S = 10$	30493,11	9590,75
Уровень языка $h = LV^*$	0,07	0,09
Количество ошибок $B = V/1000$	2	2

Таблица 6 – сводная таблица с расчетными характеристиками для программы на языке Си

3. С помощью инструмента генерации ассемблерного кода получено ассемблерное представление программы, затем в нем было произведено удаление комментариев и отладочных директив. Итоговый код представлен в Приложении В.

3.1. Произведен ручной расчет измеримых характеристик для программы на языке Си. Результат представлен в Таблице 7.

№	Оператор	Количество	№	Операнд	Количество
1	Push	3	1	Rbp	7
2	Mov	11	2	Rsp	5
3	Add	1	3	-128	1
4	Movsd	100	4	Xmm0	141
5	Call Exp	2	5	Rax	18
6	Call Erf	1	6	0	18
7	Call erfc	1	7	Xmm1	53
8	jne .L9	1	8	Xmm2	33
9	jne .L13	1	9	Xmm3	25
10	Mulsd	28	10	Xmm4	20
11	Addsd	26	11	Xmm5	8
12	Movq	14	12	Done	4
13	Xorpd	1	13	Esi	1
14	Movapd	14	14	Eax	10
15	Divsd	14	15	Er	5
16	Leave	3	16	Ec	5
17	Ret	3	17	8	5
18	sub	1	18	1	2
19	Pxor	4	19	al	1
20	Comisd	2	20	48	3
21	jbe .L20	1	21	x	9
22	jmp .L8	3	22	QWORD PTR [rbp-120]	3
23	jmp .L14	1	23	QWORD PTR .LC0	4
24	jp .L9	1	24	QWORD PTR [rbp-8]	2
25	Subsd	3	25	QWORD PTR .LC1[rip]	2

26	Movzx	2	26	QWORD PTR [rbp-16]	5
27	Test	1	27	QWORD PTR [rbp-24]	15
28	jbe .L19	1	28	QWORD PTR .LC2[rip]	1
	<b>Bcero</b>	<b>230</b>	29	QWORD PTR [rbp-32]	6
			30	QWORD PTR .LC3[rip]	1
			31	QWORD PTR [rbp-40]	5
			32	QWORD PTR .LC4[rip]	1
			33	QWORD PTR [rbp-48]	2
			34	QWORD PTR .LC5[rip]	1
			35	QWORD PTR [rbp-56]	2
			36	QWORD PTR .LC6[rip]	1
			37	QWORD PTR [rbp-64]	2
			38	QWORD PTR .LC7[rip]	1
			39	QWORD PTR [rbp-72]	2
			40	QWORD PTR .LC8[rip]	1
			41	QWORD PTR [rbp-80]	2
			42	QWORD PTR .LC9[rip]	1

			43	QWORD PTR [rbp-88]	2
			44	QWORD PTR .LC10[rip]	1
			45	QWORD PTR [rbp-96]	3
			46	QWORD PTR [rbp-120]	3
			47	QWORD PTR [rbp-104]	13
			48	QWORD PTR [rbp-112]	2
			49	QWORD PTR .LC11[rip]	1
			50	QWORD PTR .LC12[rip]	12
			51	QWORD PTR [rbp-120]	3
			52	QWORD PTR .LC13[rip]	1
			53	QWORD PTR .LC14[rip]	1
			54	QWORD PTR .LC15[rip]	1
			55	QWORD PTR .LC16[rip]	1
			56	QWORD PTR .LC17[rip]	1
			57	QWORD PTR .LC18[rip]	1
			58	QWORD PTR .LC19[rip]	1
			59	QWORD PTR .LC20[rip]	1

			60	QWORD PTR .LC21[rip]	1
			61	QWORD PTR .LC22[rip]	1
			62	QWORD PTR .LC23[rip]	1
			63	QWORD PTR x[rip]	8
			64	QWORD PTR er[rip]	4
			65	QWORD PTR .LC25[rip]	1
			66	QWORD PTR ec[rip]	4
			67	BYTE PTR done[rip]	3
				<b>Всего</b>	<b>505</b>

Таблица 7 – ручной расчет измеримых характеристик в программе на ассемблере

3.2. Расчетные характеристики вычислены вручную и с помощью программы. Результаты представлены в Таблице 8.

Характеристика	Ручной расчет
Число простых операторов $n_1$	28
Число простых операндов $n_2$	67
Общее число всех операторов $N_1$	230
Общее число всех операндов $N_2$	505
Словарь $n = n_1 + n_2$	95
Длина $N = N_1 + N_2$	735
Потенциальная теоретич. длина $N_{\text{теор}} = \sum n_i \log_2(n_i)$	541,03
Объем $V = N \log_2(n)$	4828,84
Потенциальный объем $V^* = (n_2^* + 2) \log_2(n_2^* + 2)$ , $n_2^* = 3$	11,61
Уровень $L = V^*/V$	0,002
Интеллектуальное содержание $I = 2/n_1 * n_2/N_2 * V$	45,76

Работа по программированию $E=V/L$	2008480,21
Время программирования $T=E/S$ , $S=10$	200848,02
Уровень языка $h=LV^*$	0,03
Количество ошибок $B=V/1000$	5

Таблица 8 – сводная таблица с расчетными характеристиками для программы на ассемблере

4. Сравнение результатов определения метрических характеристик представлено в Таблице 9

Характеристика	Ручной ассемблер	Ручной Си	Программный Си	Ручной Паскаль	Программный Паскаль
Число простых операторов $n_1$	28	14	11	15	15
Число простых операндов $n_2$	67	48	43	49	51
Общее число всех операторов $N_1$	230	124	128	135	158
Общее число всех операндов $N_2$	505	192	118	176	152
Словарь $n = n_1 + n_2$	95	62	54	64	66
Длина $N = N_1 + N_2$	735	316	246	311	310
Потенциальная длина $N_{\text{теор}} = \sum n_i \ln(n_i)$	541,03	321,38	271,38	333,72	347,90
Объем $V = N \ln(n)$	4828,84	1881,53	1415,70	1866	1873,76
Потенциальный объем $V^* = (n_2^* + 2) \ln(n_2^* + 2)$ , $n_2^* = 68$	11,61	11,61	11,61	11,61	11,61
Уровень $L = V^*/V$	0,002	0,006	0,008	0,006	0,006
Интеллектуальное содержание $I = 2/n_1 * n_2/N_2 * V$	45,76	67,19	93,80	69,27	83,83



Работа по программированию $E=V/L$	2008480,21	304931,08	172634	299919,36	302420
Время программирования $T=E/S, S=20$	200848,02	30493,11	9590,75	29991,94	16801.10
Уровень языка $h=LV^*$	0,03	0,07	0,09	0,07	0.07
Количество ошибок $B= V/1000$	5	2	2	2	2

Таблица 9 – сводная таблица расчетов для трех языков

По итоговой таблице видно, что Ассемблер обладает намного более низким уровнем, но при этом необходимо заметно больше работы по программированию и выше вероятность совершить ошибку.

### **Выводы**

В результате выполнения данной лабораторной работы была изучена система метрик Холстеда. Было проведено сравнение программ на языках Паскаль, Си и Ассемблер.

## Приложение А. Исходный код на Паскале

```
program erfd4;

{ evaluation of the gaussian error function }

var   x,er,ec           : real;
      done              : boolean;

function erf(x: real): real;
{ infinite series expansion of the Gaussian error function }

const sqrtpi           = 1.7724538;
      t2                = 0.66666667;
      t3                = 0.66666667;
      t4                = 0.07619048;
      t5                = 0.01693122;
      t6                = 3.078403E-3;
      t7                = 4.736005E-4;
      t8                = 6.314673E-5;
      t9                = 7.429027E-6;
      t10               = 7.820028E-7;
      t11               = 7.447646E-8;
      t12               = 6.476214E-9;

var   x2,sum            : real;

begin
  x2:=x*x;
  sum:=t5+x2*(t6+x2*(t7+x2*(t8+x2*(t9+x2*(t10+x2*(t11+x2*t12))))));
  erf:=2.0*exp(-x2)/sqrtpi*(x*(1+x2*(t2+x2*(t3+x2*(t4+x2*sum)))));
end; { function erf }

function erfc(x: real): real;
{ complement of error function }
const sqrtpi           = 1.7724538;

var   x2,v,sum          : real;

begin
  x2:=x*x;
  v:=1.0/(2.0*x2);
  sum:=v/(1+8*v/(1+9*v/(1+10*v/(1+11*v/(1+12*v)))));
  sum:=v/(1+3*v/(1+4*v/(1+5*v/(1+6*v/(1+7*sum)))));
  erfc:=1.0/(exp(x2)*x*sqrtpi*(1+v/(1+2*sum)))
end; { function erfc }

begin { main }
  done:=false;
  x:= 2.0;
  repeat
    if x<0.0 then done:=true
    else
      begin
        if x=0.0 then
          begin
            er:=0.0;
            ec:=1.0
          end
        else
          begin
            if x<1.5 then
```

```

        begin
            er:=erf(x);
            ec:=1.0-er
        end
    else
        begin
            ec:=erfc(x);
            er:=1.0-ec
        end { if }
    end;
    x := x-1;
end { if }
until done
end.

```

## Приложение Б. Исходный код на Си

```
#include <stdio.h>
#include <stdbool.h>
#include <math.h>

double erf(double x){
    const double sqrtpi      = 1.7724538;
    double t2                = 0.66666667;
    double t3                = 0.66666667;
    double t4                = 0.07619048;
    double t5                = 0.01693122;
    double t6                = 3.078403E-3;
    double t7                = 4.736005E-4;
    double t8                = 6.314673E-5;
    double t9                = 7.429027E-6;
    double t10               = 7.820028E-7;
    double t11               = 7.447646E-8;
    double t12               = 6.476214E-9;

    double x2, sum;
    x2 = x*x;
    sum = t5+x2*(t6+x2*(t7+x2*(t8+x2*(t9+x2*(t10+x2*(t11+x2*t12))))));
    return (2.0*exp(-x2)/sqrtpi*(x*(1+x2*(t2+x2*(t3+x2*(t4+x2*sum))))));
}

double erfc(double x){
    const double sqrtpi      = 1.7724538;
    double x2,v,sum;
    x2 = x*x;
    v = 1/(2*x2);
    sum=v/(1+8*v/(1+9*v/(1+10*v/(1+11*v/(1+12*v)))));
    sum=v/(1+3*v/(1+4*v/(1+5*v/(1+6*v/(1+7*sum)))));
    return (1.0/(exp(x2)*x*sqrtpi*(1+v/(1+2*sum))));
```

```

    }

int main()
{
    double x,er,ec;
    bool done;
    x = 2.0;
    done = false;
    do{
        if(x<0){
            done = true;
        }else if (x == 0){
            er = 0;
            ec = 1;
        }else if (x < 1.5){
            er = erf(x);
            ec = 1 - er;
        }else{
            ec = erfc(x);
            er = 1-ec;
        }
        x = x - 1;

    }while (done == false);

    return 0;
}

```

## Приложение В. Исходный код на ассемблере

```
x:
    .zero 8
er:
    .zero 8
ec:
    .zero 8
done:
    .zero 1
erf:
    push    rbp
    mov     rbp, rsp
    add     rsp, -128
    movsd   QWORD PTR [rbp-120], xmm0
    movsd   xmm0, QWORD PTR .LC0[rip]
    movsd   QWORD PTR [rbp-8], xmm0
    movsd   xmm0, QWORD PTR .LC1[rip]
    movsd   QWORD PTR [rbp-16], xmm0
    movsd   xmm0, QWORD PTR .LC1[rip]
    movsd   QWORD PTR [rbp-24], xmm0
    movsd   xmm0, QWORD PTR .LC2[rip]
    movsd   QWORD PTR [rbp-32], xmm0
    movsd   xmm0, QWORD PTR .LC3[rip]
    movsd   QWORD PTR [rbp-40], xmm0
    movsd   xmm0, QWORD PTR .LC4[rip]
    movsd   QWORD PTR [rbp-48], xmm0
    movsd   xmm0, QWORD PTR .LC5[rip]
    movsd   QWORD PTR [rbp-56], xmm0
    movsd   xmm0, QWORD PTR .LC6[rip]
    movsd   QWORD PTR [rbp-64], xmm0
    movsd   xmm0, QWORD PTR .LC7[rip]
    movsd   QWORD PTR [rbp-72], xmm0
    movsd   xmm0, QWORD PTR .LC8[rip]
    movsd   QWORD PTR [rbp-80], xmm0
```

```

movsd xmm0, QWORD PTR .LC9[rip]
movsd QWORD PTR [rbp-88], xmm0
movsd xmm0, QWORD PTR .LC10[rip]
movsd QWORD PTR [rbp-96], xmm0
movsd xmm0, QWORD PTR [rbp-120]
mulsd xmm0, xmm0
movsd QWORD PTR [rbp-104], xmm0
movsd xmm0, QWORD PTR [rbp-104]
mulsd xmm0, QWORD PTR [rbp-96]
addsd xmm0, QWORD PTR [rbp-88]
mulsd xmm0, QWORD PTR [rbp-104]
addsd xmm0, QWORD PTR [rbp-80]
mulsd xmm0, QWORD PTR [rbp-104]
addsd xmm0, QWORD PTR [rbp-72]
mulsd xmm0, QWORD PTR [rbp-104]
addsd xmm0, QWORD PTR [rbp-64]
mulsd xmm0, QWORD PTR [rbp-104]
addsd xmm0, QWORD PTR [rbp-56]
mulsd xmm0, QWORD PTR [rbp-104]
addsd xmm0, QWORD PTR [rbp-48]
mulsd xmm0, QWORD PTR [rbp-104]
movsd xmm1, QWORD PTR [rbp-40]
addsd xmm0, xmm1
movsd QWORD PTR [rbp-112], xmm0
movsd xmm0, QWORD PTR [rbp-104]
movq xmm1, QWORD PTR .LC11[rip]
xorpd xmm0, xmm1
movq rax, xmm0
movq xmm0, rax
call exp
addsd xmm0, xmm0
movsd xmm2, QWORD PTR .LC0[rip]
movapd xmm1, xmm0
divsd xmm1, xmm2

```

```

movsd  xmm0, QWORD PTR [rbp-104]
mulsd  xmm0, QWORD PTR [rbp-112]
addsd  xmm0, QWORD PTR [rbp-32]
mulsd  xmm0, QWORD PTR [rbp-104]
addsd  xmm0, QWORD PTR [rbp-24]
mulsd  xmm0, QWORD PTR [rbp-104]
addsd  xmm0, QWORD PTR [rbp-16]
movapd  xmm2, xmm0
mulsd  xmm2, QWORD PTR [rbp-104]
movsd  xmm0, QWORD PTR .LC12[rip]
addsd  xmm0, xmm2
mulsd  xmm0, QWORD PTR [rbp-120]
mulsd  xmm0, xmm1
movq    rax, xmm0
movq    xmm0, rax
leave
ret

```

erfc:

```

push    rbp
mov     rbp, rsp
sub     rsp, 48
movsd   QWORD PTR [rbp-40], xmm0
movsd   xmm0, QWORD PTR .LC0[rip]
movsd   QWORD PTR [rbp-8], xmm0
movsd   xmm0, QWORD PTR [rbp-40]
mulsd   xmm0, xmm0
movsd   QWORD PTR [rbp-16], xmm0
movsd   xmm0, QWORD PTR [rbp-16]
movapd  xmm1, xmm0
addsd   xmm1, xmm0
movsd   xmm0, QWORD PTR .LC12[rip]
divsd   xmm0, xmm1
movsd   QWORD PTR [rbp-24], xmm0
movsd   xmm1, QWORD PTR [rbp-24]

```



```

movsd xmm0, QWORD PTR .LC13[rip]
mulsd xmm0, xmm1
movsd xmm2, QWORD PTR [rbp-24]
movsd xmm1, QWORD PTR .LC14[rip]
mulsd xmm1, xmm2
movsd xmm3, QWORD PTR [rbp-24]
movsd xmm2, QWORD PTR .LC15[rip]
mulsd xmm2, xmm3
movsd xmm4, QWORD PTR [rbp-24]
movsd xmm3, QWORD PTR .LC16[rip]
mulsd xmm3, xmm4
movsd xmm5, QWORD PTR [rbp-24]
movsd xmm4, QWORD PTR .LC17[rip]
mulsd xmm5, xmm4
movsd xmm4, QWORD PTR .LC12[rip]
addsd xmm5, xmm4
movapd xmm4, xmm3
divsd xmm4, xmm5
movsd xmm3, QWORD PTR .LC12[rip]
addsd xmm4, xmm3
movapd xmm3, xmm2
divsd xmm3, xmm4
movsd xmm2, QWORD PTR .LC12[rip]
addsd xmm3, xmm2
movapd xmm2, xmm1
divsd xmm2, xmm3
movsd xmm1, QWORD PTR .LC12[rip]
addsd xmm2, xmm1
movapd xmm1, xmm0
divsd xmm1, xmm2
movsd xmm0, QWORD PTR .LC12[rip]
addsd xmm1, xmm0
movsd xmm0, QWORD PTR [rbp-24]
divsd xmm0, xmm1

```

```

movsd  QWORD PTR [rbp-32], xmm0
movsd  xmm1, QWORD PTR [rbp-24]
movsd  xmm0, QWORD PTR .LC18[rip]
mulsd  xmm0, xmm1
movsd  xmm2, QWORD PTR [rbp-24]
movsd  xmm1, QWORD PTR .LC19[rip]
mulsd  xmm1, xmm2
movsd  xmm3, QWORD PTR [rbp-24]
movsd  xmm2, QWORD PTR .LC20[rip]
mulsd  xmm2, xmm3
movsd  xmm4, QWORD PTR [rbp-24]
movsd  xmm3, QWORD PTR .LC21[rip]
mulsd  xmm3, xmm4
movsd  xmm5, QWORD PTR [rbp-32]
movsd  xmm4, QWORD PTR .LC22[rip]
mulsd  xmm5, xmm4
movsd  xmm4, QWORD PTR .LC12[rip]
addsd  xmm5, xmm4
movapd xmm4, xmm3
divsd  xmm4, xmm5
movsd  xmm3, QWORD PTR .LC12[rip]
addsd  xmm4, xmm3
movapd xmm3, xmm2
divsd  xmm3, xmm4
movsd  xmm2, QWORD PTR .LC12[rip]
addsd  xmm3, xmm2
movapd xmm2, xmm1
divsd  xmm2, xmm3
movsd  xmm1, QWORD PTR .LC12[rip]
addsd  xmm2, xmm1
movapd xmm1, xmm0
divsd  xmm1, xmm2
movsd  xmm0, QWORD PTR .LC12[rip]
addsd  xmm1, xmm0

```

```

movsd xmm0, QWORD PTR [rbp-24]
divsd xmm0, xmm1
movsd QWORD PTR [rbp-32], xmm0
mov rax, QWORD PTR [rbp-16]
movq xmm0, rax
call exp
movq rax, xmm0
movq xmm1, rax
mulsd xmm1, QWORD PTR [rbp-40]
movsd xmm0, QWORD PTR .LC0[rip]
mulsd xmm1, xmm0
movsd xmm0, QWORD PTR [rbp-32]
movapd xmm2, xmm0
addsd xmm2, xmm0
movsd xmm0, QWORD PTR .LC12[rip]
movapd xmm3, xmm2
addsd xmm3, xmm0
movsd xmm0, QWORD PTR [rbp-24]
movapd xmm2, xmm0
divsd xmm2, xmm3
movsd xmm0, QWORD PTR .LC12[rip]
addsd xmm0, xmm2
mulsd xmm1, xmm0
movsd xmm0, QWORD PTR .LC12[rip]
divsd xmm0, xmm1
movq rax, xmm0
movq xmm0, rax
leave
ret

```

main:

```

push rbp
mov rbp, rsp
movsd xmm0, QWORD PTR .LC23[rip]
movsd QWORD PTR x[rip], xmm0

```

```

        mov     BYTE PTR done[rip], 0
.L14:
        movsd   xmm1, QWORD PTR x[rip]
        pxor    xmm0, xmm0
        comisd   xmm0, xmm1
        jbe     .L19
        mov     BYTE PTR done[rip], 1
        jmp     .L8
.L19:
        movsd   xmm0, QWORD PTR x[rip]
        pxor    xmm1, xmm1
        ucomisd  xmm0, xmm1
        jp      .L9
        pxor    xmm1, xmm1
        ucomisd  xmm0, xmm1
        jne     .L9
        pxor    xmm0, xmm0
        movsd   QWORD PTR er[rip], xmm0
        movsd   xmm0, QWORD PTR .LC12[rip]
        movsd   QWORD PTR ec[rip], xmm0
        jmp     .L8
.L9:
        movsd   xmm1, QWORD PTR x[rip]
        movsd   xmm0, QWORD PTR .LC25[rip]
        comisd   xmm0, xmm1
        jbe     .L20
        mov     rax, QWORD PTR x[rip]
        movq     xmm0, rax
        call    erf
        movq     rax, xmm0
        mov     QWORD PTR er[rip], rax
        movsd   xmm1, QWORD PTR er[rip]
        movsd   xmm0, QWORD PTR .LC12[rip]
        subsd   xmm0, xmm1

```

```
    movsd  QWORD PTR ec[rip], xmm0
    jmp    .L8
```

.L20:

```
    mov    rax, QWORD PTR x[rip]
    movq   xmm0, rax
    call   erfc
    movq   rax, xmm0
    mov    QWORD PTR ec[rip], rax
    movsd  xmm1, QWORD PTR ec[rip]
    movsd  xmm0, QWORD PTR .LC12[rip]
    subsd  xmm0, xmm1
    movsd  QWORD PTR er[rip], xmm0
```

.L8:

```
    movsd  xmm0, QWORD PTR x[rip]
    movsd  xmm1, QWORD PTR .LC12[rip]
    subsd  xmm0, xmm1
    movsd  QWORD PTR x[rip], xmm0
    movzx  eax, BYTE PTR done[rip]
    movzx  eax, al
    test   eax, eax
    jne    .L13
    jmp    .L14
```

.L13:

```
    mov    eax, 0
    pop    rbp
    ret
```