

Министерство образования и науки РФ  
Санкт-Петербургский политехнический университет Петра Великого  
Институт компьютерных наук и технологий  
Высшая школа программной инженерии

Отчёт по лабораторной работе №1  
по дисциплине «Вычислительная математика»

**Вариант №15**

Выполнил

Руководитель

Воскобойников С.П.

## **Оглавление**

Оглавление .....	2
Описание работы .....	3
1. Постановка задачи: .....	3
2. Текст программы .....	3
3. Результаты работы программы .....	3
4. Выводы по результатам .....	6

# Описание работы

## 1. Постановка задачи:

Для  $2 \leq x \leq 3$  с шагом  $h = 0.1$  вычислить значения функции  $f(x)$  с использованием программы QUANC8, где  $f(x) = \int_0^x \frac{1 - \cos(t)}{t} dt$ . По полученным точкам построить сплайн-функцию и полином Лагранжа 10-й степени. В точках  $x_k = 2.05 + 0.1k$  для  $k = 0, 1, \dots, 9$  сравнить значения сплайн-функции и полинома с точным значением  $f(x)$  (вычислить интеграл по QUANC8 с высокой точностью).

## 2. Текст программы

```
#include <math.h>
#include <stdio.h>
#include "cmath.h"

// SPLINE values
#define ndim 11
double x[ndim], y[ndim], bspl[ndim], cspl[ndim], dspl[ndim], u[ndim], fspline[ndim];

// Lagrange values
double temp[ndim];

// Total comparison values
double quancArr[ndim], splineArr[ndim], lagrangeArr[ndim], splineDiff, lagrangeDiff;

double fquanc(double t) // integrand (fun)
{
    double temp;
    if (t == 0.0) temp = 0.0;
    if (t != 0.0) temp = (1 - cos(t)) / t;
    return (temp);
}

void quancer(double a, double b, double bx, double step) // QUANC8 function for interval
{
    double result, errest, posn;
    int nfe, flag;
    double epsrel = 1.0e-10; // relative epsilon
    double epsabs = 0.0; // absolute epsilon
    int i = 0;
    while(b < bx + step)
    {
        quanc8(fquanc, a, b, epsabs, epsrel, &result, &errest, &nfe, &posn, &flag);
        x[i] = b;
        y[i] = result;
        b += step;
        i++;
    }
}

void splinerXk(double n, double final, double step) // U array creation for SPLINE
{
    int i = 0;
    while (n < final + step)
    {
        u[i] = n;
        n += step;
        i++;
    }
}
```

```

}

void spliner() // SPLINE function for interval, uses QUANC8 calculations
{
    int last, flagspline;
    spline(ndim, 1, 1, 1, x, y, bspl, cspl, dspl, &flagspline);
    for (int i = 0; i < ndim; ++i)
    {
        fspline[i] = seval(ndim, u[i], x, y, bspl, cspl, dspl, &last);
    }
}

double lagranger(double xlag) // Lagrange function for interval, uses QUANC8 calculations
{
    double resultlag = 0;
    double numer[ndim], denom[ndim];
    for (int i = 0; i < ndim; i++)
    {
        for (int j = 0; j < ndim; j++)
        {
            if (i != j)
            {
                numer[i] *= xlag - x[j];
                denom[i] *= x[i] - x[j];
            }
        }
        if (denom[i] != 0)
        {
            resultlag += (numer[i] / denom[i]) * y[i];
        }
    }
    return resultlag;
}

void totalComp(double a, double b, double bx, double step, double bp, double bxp, double
stepp) // the overall comparison
{
    printf("-----\n");
    printf("|                %.2f <= x <= %.2f                |\n", b, bx);
    printf("-----\n");
    printf("|                x                |                QUANC8                |\n");
    printf("-----\n");
    quancer(a, b, bx, step);
    for (int i = 0; i < ndim; i++)
    {
        quancArr[i] = y[i];
        printf("|%17.2f                |%22.9f                |\n", x[i], quancArr[i]);
    }
    splinerXk(2.05, 2.95, 0.1);
    spliner();
    for (int z = 0; z < ndim - 1; z++)
    {
        temp[z] = lagranger(u[z]);
    }
    quancer(0, bp, bxp, stepp);
    printf("-----\n");
    printf("|                %.2f <= x <= %.2f                |\n", 2.05,
2.95);
    printf("-----\n");
    printf("|    x    |    QUANC8    |    SPLINE    |    Lagrange    |\n");
    printf("-----\n");
    for (int i = 0; i < ndim - 1; i++)
    {
        quancArr[i] = y[i];
        splineArr[i] = fspline[i];
    }
}

```

```

    lagrangeArr[i] = temp[i];
    printf("|%7.2f    |%13.9f    |%13.9f    |%14.9f    |\n", u[i], quancArr[i],
splineArr[i], lagrangeArr[i]);
}
printf("-----\n");
printf("|      X      |      QUANC8      |      Q8 - S      |      Q8 - Lgr      |\n");
printf("-----\n");
for (int i = 0; i < ndim - 1; i++)
{
    splineDiff = fabs(quancArr[i] - splineArr[i]);
    lagrangeDiff = fabs(quancArr[i] - lagrangeArr[i]);
    printf("|%7.2f    |%13.9f    |%14.6e    |%15.6e    |\n", x[i], quancArr[i], splineDiff,
lagrangeDiff);
}
printf("-----\n");
printf("|                                     |\n");
printf("-----\n\n");
}

int main(void)
{
    totalComp(0, 2.0, 3.0, 0.1, 2.05, 2.95, 0.1);
    return (0);
}

```

### 3. Результаты работы программы

2.00 <= x <= 3.00			
X	QUANC8		
2.00	0.847382017		
2.10	0.918641022		
2.20	0.990598426		
2.30	1.062949170		
2.40	1.135392785		
2.50	1.207635200		
2.60	1.279390494		
2.70	1.350382559		
2.80	1.420346692		
2.90	1.489031078		
3.00	1.556198168		
2.05 <= x <= 2.95			
X	QUANC8	SPLINE	Lagrange
2.05	0.882905024	0.887531900	0.882905024
2.15	0.954551576	0.953311666	0.954551576
2.25	1.026743539	1.027076246	1.026743539
2.35	1.099177919	1.099086914	1.099177919
2.45	1.171557224	1.171588421	1.171557224
2.55	1.243591243	1.243557319	1.243591243
2.65	1.314998759	1.315103093	1.314998759
2.75	1.385509175	1.385125574	1.385509175
2.85	1.454864050	1.456293907	1.454864050
2.95	1.522818539	1.517482479	1.522818539
X	QUANC8	Q8 - S	Q8 - Lgr
2.05	0.882905024	4.626876e-03	7.438494e-15
2.15	0.954551576	1.239910e-03	1.554312e-15
2.25	1.026743539	3.327071e-04	2.220446e-16
2.35	1.099177919	9.100454e-05	4.440892e-16
2.45	1.171557224	3.119699e-05	4.440892e-16
2.55	1.243591243	3.392424e-05	2.220446e-16
2.65	1.314998759	1.043337e-04	6.661338e-16
2.75	1.385509175	3.836008e-04	2.220446e-16
2.85	1.454864050	1.429857e-03	8.881784e-16
2.95	1.522818539	5.336060e-03	6.439294e-15

#### **4. Выводы по результатам**

По результатам, полученным в ходе работы программы, мы видим, что сплайн-функция и полином Лагранжа описывают функцию практически одинаково, однако вторая аппроксимация оказалась точнее.