

Лабораторная работа № 1

1. Тема: численное интегрирование

2. Постановка задачи:

Составить программу, которая реализует методы численного интегрирования с постоянным и переменным шагом для интеграла

$$\int_0^1 e^{-x^2} dx$$



3. Мат. модель:

1. Метод прямоугольников левых частей:

$$\int_a^b f(x) dx \approx h * \sum_{x=a}^{b-h} f(x), \text{ где } h = \frac{b-a}{n}, f(x) = e^{-x^2}$$

2. Метод прямоугольников правых частей:

$$\int_a^b f(x) dx \approx h * \sum_{x=a+h}^b f(x), \text{ где } h = \frac{b-a}{n}, f(x) = e^{-x^2}$$

3. Метод трапеций

$$\int_a^b f(x) dx \approx h * \left(\frac{f(a) + f(b)}{2} + \sum_{x=a+h}^{b-h} f(x) \right), \text{ где } h = \frac{b-a}{n}, f(x) = e^{-x^2}$$

4. Метод парабол

$$\int_a^b f(x) dx \approx \frac{h}{3} * (f(a) + 4 * \sum_{x=a+h}^{b-h} f(x) + 2 * \sum_{x=a+2*h}^{b-2*h} f(x) + f(b)),$$

$$\text{где } h = \frac{(b-a)}{n}, f(x) = e^{-x^2}$$

5. Алгоритм 1 – используется метод прямоугольников левых частей
6. Алгоритм 2 – используется метод прямоугольников левых частей
7. Остаточный член для метода прямоугольников

$$|R| \leq \frac{(b-a)^2}{2n} * M, M = \max (|(f(x))'|)$$

8. Остаточный член для метода трапеций

$$|R| \leq \frac{(b-a)^3}{12n^2} * M, M = \max (|(f(x))''|)$$

9. Остаточный член для метода парабол

$$|R| \leq \frac{(b-a)^5}{120(2n)^4} * M, M = \max (|(f(x))^{IV}|)$$

4. Список идентификаторов: (в скобках указаны функции, в которых находится переменная)

| Имя | Тип | Смысл |
|-----|-------|---|
| a | const | Левый предел интегрирования |
| b | const | Правый предел интегрирования |
| xf | float | Переменная для функции с вычисляемым подынтегральным выражением (f) |
| R1 | int | Переменная для выбора остаточного члена определенного метода (ErrorTerm) |
| a2 | float | Левый предел интегрирования (ErrorTerm) |
| b2 | float | Правый предел интегрирования (ErrorTerm) |
| n2 | int | Количество частей разбиения (ErrorTerm) |
| m1 | float | Разность пределов интегрирования (ErrorTerm) |
| M | float | Максимальное из модулей от производных подынтегральной функции (ErrorTerm) |
| a1 | float | Левый предел интегрирования, который передается в функции (LeftRiemannSum, RightRiemannSum, TrapezoidalRule, SimpsonsRule, Algorithm1, Algorithm2) |
| b1 | float | Правый предел интегрирования, который передается в функции (LeftRiemannSum, RightRiemannSum, TrapezoidalRule, SimpsonsRule, Algorithm1, Algorithm2) |
| n1 | float | Количество частей разбиения, которое передается в функции (LeftRiemannSum, RightRiemannSum, TrapezoidalRule, SimpsonsRule, Algorithm1, Algorithm2) |
| S | float | Сумма для вычисления интеграла (LeftRiemannSum, RightRiemannSum, TrapezoidalRule) |
| h | float | Шаг для вычисления (LeftRiemannSum, RightRiemannSum, TrapezoidalRule, SimpsonsRule, Algorithm1) |
| x | float | Начальное значение (LeftRiemannSum, RightRiemannSum, TrapezoidalRule, SimpsonsRule, Algorithm1, Algorithm2) |
| I | float | Значение интеграла (LeftRiemannSum, RightRiemannSum, TrapezoidalRule, SimpsonsRule) |

| | | |
|--------|-------|--|
| fa, fb | float | Значение функции в точках a и b соответственно (TrapezoidalRule, SimpsonsRule) |
| S1, S2 | float | Суммы для вычисления в SimpsonsRule |
| e | float | Точность вычисления (Algorithm1, Algorithm2) |
| R | float | Остаточный член (Algorithm1, Algorithm2) |
| In | float | Интеграл с шагом h (Algorithm1) |
| S2 | float | Сумма для вычисления интеграла (Algorithm1) |
| I2n | float | Интеграл с шагом h/2 (Algorithm1) |
| hv | float | Шаг вычисления (Algorithm2) |
| S1 | float | Сумма для вычисления интеграла (Algorithm2) |
| hd | float | Шаг движения (Algorithm2) |
| hs | float | Шаг сдвига (Algorithm2) |
| k | int | Переменная для выбора нужного меню (MainMenuChoose) |
| p | int | Переменная для ввода (MainMenuChoose) |
| t | int | Переменная для выбора нужного действия после подсчета интеграла (Choose) |
| n2 | int | Переменная количества разбиений, вводится с клавиатуры (Parts) |
| k2 | int | Переменная для выбора в меню с методами (MenuChoose) |
| n | int | Переменная для выбора действия после вычисления (MenuChoose) |
| i | int | Выбор меню (main) |

5. Код программы:

```
#include <iostream>

#include <stdlib.h>

#include <math.h>

#include <conio.h>

#define a 0

#define b 1

using namespace std;

int MainMenu(){ //Функция для вывода главного меню

    cout << "Главное меню:" << endl;

    cout << "1. Методы с постоянным шагом" << endl;

    cout << "2. Методы с переменным шагом" << endl;

    cout << "3. Выход из программы" << endl;
```

```
}
```

```
int Output1(){ //Функция для вывода меню "Методы с постоянным шагом"  
    cout << "Методы с постоянным шагом:" << endl;  
    cout << "1. Метод правых частей прямоугольников" << endl;  
    cout << "2. Метод левых частей прямоугольников" << endl;  
    cout << "3. Метод трапеций" << endl;  
    cout << "4. Метод парабол" << endl;  
    cout << "5. Возврат в главное меню" << endl;  
}
```

```
int Output2(){ //Функция для вывода меню "Методы с переменным шагом"  
    cout << "Методы с постоянным шагом:" << endl;  
    cout << "1. Метод по 1 алгоритму" << endl;  
    cout << "2. Метод по 2 алгоритму" << endl;  
    cout << "3. Возврат в главное меню" << endl;  
}
```

```
float f(float xf){ //Функция с вычисляемым подынтегральным выражением  
    return exp(-xf*xf);  
}
```

```
int ErrorTerm(int R1, float a2, float b2, int n2){ //Функция вычисления  
остаточного члена для разных методов  
    float m1 = b2 - a2, M;  
    switch(R1){  
    case 1: { //Метод прямоугольников  
        M = max(fabs(f(a2)*(-2*a2)), fabs(f(b2)*(-2*b2)));  
        return m1*m1*M/(2*n2);  
    }  
}
```

case 2: { //Метод трапеций

$M = \max(\text{fabs}(af(a2)*(4*a2*a2 - 2)), \text{fabs}(f(b2)*(4*b2*b2 - 2)));$

$\text{return } m1*m1*m1*M/(12*n2*n2);$

}

case 3: { //Метод парабол

$M = \max(\text{fabs}(f(a2)*(12*pow(a2, 4) + 6*a2 + 12)), \text{fabs}(f(b2)*(12*pow(b2, 4) + 6*b2 + 12)));$

$\text{return } pow(m1, 5)*M/(180*pow(2*n2, 4));$

}

}

}

float LeftRiemannSum(float a1, float b1, float n1){ //Метод прямоугольников
левых частей

$\text{float } S = 0, h = (b1 - a1) / n1, x = a1, I;$

$\text{while } (x \leq b1 - h)\{$

$S = S + f(x);$

$x = x + h;$

}

$I = S*h;$

$\text{return } I + \text{ErrorTerm}(1, a1, b1, n1);$

}

float RightRiemannSum(float a1, float b1, float n1){ //Метод прямоугольников
правых частей

$\text{float } S = 0, h = (b1 - a1) / n1, x = a1 + h, I;$

$\text{while } (x \leq b1)\{$

$S = S + f(x);$

$x = x + h;$

}

```

    I = S*h;
    return I + ErrorTerm(1, a1, b1, n1);
}

```

```

float TrapezoidalRule(float a1, float b1, float n1){ //Метод трапеций
    float S = 0, h = (b1 - a1)/n1, fa = f(a1), fb = f(b1), x = a1 + h, I;
    while (x <= b1 - h){
        S = S + f(x);
        x = x + h;
    }
    I = (fa/2 + fb/2 + S)*h;
    return I + ErrorTerm(2, a1, b1, n1);
}

```

```

float SimpsonsRule(float a1, float b1, float n1){ //Метод парабол
    float S1 = 0, S2 = 0, h = (b1 - a1) / n1, fa = f(a1), fb = f(b1), x = a1 + h, I;
    while (x <= b1 - h){
        S1 = S1 + f(x);
        x = x + 2*h;
    }
    x = a1 + 2*h;
    while (x <= b1 - 2*h){
        S2 = S2 + f(x);
        x = x + 2*h;
    }
    I = h*(fa + 4*S1 + 2*S2 + fb)/3;
    return I + ErrorTerm(3, a1, b1, n1);
}

```

```

float Algorithm1(float a1, float b1, float n1){ //Алгоритм 1
    float e = pow(10, -5), h = (b1 - a1)/n1, In = 0, R = ErrorTerm(1, a1, b1, n1), S2,
x, I2n;
    while (R < e){
        S2 = 0;
        x = a1;
        while(x <= b1 - h){
            S2 += f(x);
            x += h;
        }
        I2n = h*S2;
        R = fabs(In - I2n);
        In = I2n;
        h /= 2;
    }
    return In;
}

```

```

float Algorithm2(float a1, float b1, float n1){ //Алгоритм 2
    float e = pow(10, -5), hv = (b1 - a1)/n1, In = 0, R = ErrorTerm(1, a1, b1, n1),
S2, x = a1, I2n = 0;
    float S1 = f(a1), hd = hv, hs = hv;
    while (R < e){
        S2 = 0;
        x += hs;
        S2 += exp(-x*x);
        while (x <= b1 - hs){
            x += hd;
            S2 += f(x);
        }
    }
}

```

```

    I2n = hs*S1 + hd*S2;
    R = fabs(In - I2n);
    In = I2n;
    hd = hs;
    hs /= 2;
}
return In;
}

int MainMenuChoose(int k){ //Выбор в главном меню
    int p;
    switch (k) {
        case 1:{ //Методы с постоянным шагом
            Output1(); //Вывод меню
            k *= 10;
            cout << "\nВведите число от 1 до 5: "; cin >> p;
            system("cls");
            return (k + p);
            break;
        }
        case 2:{ //Методы с переменным шагом
            Output2(); //Вывод меню
            k *= 10;
            cout << "\nВведите число от 1 до 3: "; cin >> p;
            system("cls");
            return (k + p);
            break;
        }
        case 3:{ //Выход из программы

```



```

        return -1;
        break;
    }
}

```

```

int Choose(){ //Выбор после того, как выбранный интеграл посчитан
    int t;
    cout << "\n1. Вернуться в главное меню\n2. Выйти из программы" << endl;
    cout << "\nВведите число от 1 до 2: "; cin >> t;
    if (t == 1){
        t = 0;
        system("cls");
    } else {
        t = -1;
        system("cls");
    };
    return t;
}

```

```

int Parts(int z){ //Проверка, нужно ли вводить части или выйти из
программы/вернуться в главное меню
    int n2;
    if (z != -1){
        if (z != 15){
            if (z != 23){
                cout << "Введите количество частей: "; cin >> n2;
            }
        }
    }
    return n2;
}

```

```
}
```

```
int MenuChoose(int k2){ //Выбор в меню с постоянным/переменным шагом  
    int n;  
    switch (k2) {  
        case 11: { //Метод правых частей прямоугольников  
            cout << "Метод правых частей прямоугольников" << endl;  
            n = Parts(k2); //Проверка, нужно ли вводить части или выйти из  
программы/вернуться в главное меню  
            cout << "Интеграл равен: " << RightRiemannSum(a, b, n) << endl;  
            return Choose(); //Выбор после того, как выбранный интеграл посчитан  
            break;  
        }  
        case 12: { //Метод левых частей прямоугольников  
            cout << "Метод левых частей прямоугольников" << endl;  
            n = Parts(k2); //Проверка, нужно ли вводить части или выйти из  
программы/вернуться в главное меню  
            cout << "Интеграл равен: " << LeftRiemannSum(a, b, n) << endl;  
            return Choose(); //Выбор после того, как выбранный интеграл посчитан  
            break;  
        }  
        case 13: { //Метод трапеций  
            cout << "Метод трапеций" << endl;  
            n = Parts(k2); //Проверка, нужно ли вводить части или выйти из  
программы/вернуться в главное меню  
            cout << "Интеграл равен: " << TrapezoidalRule(a, b, n) << endl;  
            return Choose(); //Выбор после того, как выбранный интеграл посчитан  
            break;  
        }  
        case 14: { //Метод парабол
```

```

    cout << "Метод парабол" << endl;

    n = Parts(k2); //Проверка, нужно ли вводить части или выйти из
программы/вернуться в главное меню

    cout << "Интеграл равен: " << SimpsonsRule(a, b, n) << endl;

    return Choose(); //Выбор после того, как выбранный интеграл посчитан
break;
}

case 15: { //Возврат в главное меню
    return 0;

    system("cls");

    break;
}

case 21: { //Метод по 1 алгоритму
    cout << "Метод по 1 алгоритму" << endl;

    n = Parts(k2); //Проверка, нужно ли вводить части или выйти из
программы/вернуться в главное меню

    cout << "Интеграл равен: " << Algorithm1(a, b, n) << endl;

    return Choose(); //Выбор после того, как выбранный интеграл посчитан
break;
}

case 22: { //Метод по 2 алгоритму
    cout << "Метод по 2 алгоритму" << endl;

    n = Parts(k2); //Проверка, нужно ли вводить части или выйти из
программы/вернуться в главное меню

    cout << "Интеграл равен: " << Algorithm2(a, b, n) << endl;

    return Choose(); //Выбор после того, как выбранный интеграл посчитан
break;
}

case 23: { //Возврат в главное меню
    return 0;

```

```
        system("cls");
        break;
    }
}
}
```

```
int main(){
    system("chcp 1251 > nul");
    int i = 0;
    while (i != -1){
        MainMenu();
        cout << "\nВведите число от 1 до 3: "; cin >> i;
        system("cls");
        i = MainMenuChoose(i);
        if (i != -1){
            float n;
            i = MenuChoose(i);
        }
    }
    system("pause > nul");
    return 0;
}
```

Результаты:

