МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА» (БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения вычислительной техники и
автоматизированных систем

Лабораторная работа №3

по дисциплине: «Вычислительная математика»

тема: Численное интегрирование

Выполнил: ст. группы ПВ-211

Чувилко Илья Романович

Проверил:

Бондаренко Татьяна Владимировна

Белгород 2023 г.

Вариант 23

Цель работы: изучить понятие квадратурной формулы; изучить основные способы численного интегрирования; получить практические навыки решения задачи численного интегрирования с помощью ЭВМ.

Ход работы:

Задание 1.	23	$\int_{-2}^{1}(6x^2-2x+3)dx$	$\int_{0}^{\pi}\sin x\cdot\cos x dx$	$\int_{1}^{2}(1+\ln(2x))dx$

Вычислить «вручную» интегралы из таблицы вариантов заданий: − точно (все 3 интеграла);

− по

$$1)\ \int_{-2}^{1}(6x^2-2x+3)dx=\int_{-2}^{1}6x^2dx-\int_{-2}^{1}2xdx+\int_{-2}^{1}3dx=2x^3\Big|_{-2}^{1}-x^2\Big|_{-2}^{1}+3x\Big|_{-2}^{1}=30$$

$$2)\ \int_{0}^{\pi}\sin x\cos x dx=\int_{0}^{\pi}\frac{\sin 2x}{2}dx=\frac{1}{4}\int_{0}^{\pi}\sin 2x\, d2x=\frac{1}{4}(-\cos 2x)\Big|_{0}^{\pi}=\frac{1}{4}(-\cos(2\pi)+\cos(0))=0$$

$$3)\ \int_{1}^{2}(1+\ln(2x))dx=\int_{1}^{2}dx+\int_{1}^{2}\ln(2x)dx=\left[t=2x\ \middle|\ \begin{matrix}x=\frac{t}{2}\\ dx=\frac{1}{2}dt\end{matrix}\right]=x\Big|_{1}^{2}+\int_{1}^{2}\frac{\ln t}{2}dt=\left[\begin{matrix}u=\ln t\\ v'=1\end{matrix}\ \middle|\ \begin{matrix}u'=\frac{1}{t}\\ v=t\end{matrix}\right]=$$

$$1+\frac{1}{2}\left(t\ln t-\int dt\right)=1+\frac{1}{2}\left(2x\ln 2x\Big|_{1}^{2}-2x\Big|_{1}^{2}\right)=\frac{4\ln 4-2\ln 2}{2}\approx 2{,}08$$

формуле центральных (средних) прямоугольников, используя для оценки точности двойной просчёт при n1=8; n2=10 (интеграл 1);

− по

n	0	1	2	3	4	5	6	7		
x	-1,8125	-1,4375	-1,0625	-0,6875	-0,3125	0,0625	0,4375	0,8125		
y	26,33594	18,27344	11,89844	7,210938	4,210938	2,898438	3,273438	5,335938		
							Сумма:	29,78906		
n	0	1	2	3	4	5	6	7	8	9
x	-1,85	-1,55	-1,25	-0,95	-0,65	-0,35	-0,05	0,25	0,55	0,85
y	27,24	20,52	14,88	10,32	6,84	4,44	3,12	2,88	3,72	5,64
									Сумма:	29,865

формуле трапеций при n=8 (интеграл 1 и 2);

0	1	2	3	4	5	6	7	8
-2	-1,625	-1,25	-0,875	-0,5	-0,125	0,25	0,625	1
31	22,09375	14,875	9,34375	5,5	3,34375	2,875	4,09375	7
								30,421875
0	1	2	3	4	5	6	7	8
0	0,392699082	0,785398163	1,178097245	1,570796327	1,963495408	2,35619449	2,748893572	3,141592654
0	0,353553391	0,5	0,353553391	6,12323E-17	-0,35355339	-0,5	-0,35355339	-1,22465E-16
								6,53975E-17

− по формуле парабол (Симпсона) при n=8 (интеграл 1 и 3).

0	1	2	3	4	5	6	7	8
-2	-1,625	-1,25	-0,875	-0,5	-0,125	0,25	0,625	1
31	22,09375	14,875	9,34375	5,5	3,34375	2,875	4,09375	7
							Сумма:	30
0	1	2	3	4	5	6	7	8
1	1,125	1,25	1,375	1,5	1,625	1,75	1,875	2
1,693147181	1,810930216	1,916290732	2,011600912	2,098612289	2,178654996	2,252762968	2,32175584	2,386294361
							Сумма:	2,079439224

Замечание. Для вычисления значений «вручную» рекомендуется использовать

Microsoft Excel или другую программу

Задание 2. Определить погрешность вычисления интеграла 1 по каждой из формул. Результаты представить в виде табл. 3.1.

Погрешности	Формула центральных прямоугольников		Формула трапеций	Формула парабол n = 8	Формула Гаусса			
	n = 8	n = 10	n = 8	8	n = 1	n = 2	n = 3	n = 4
Δ1	0,2109375	0,135	0,421875	0	12,375	0	0,0005	0,0001
δ1	0,00703125	0,0045	0,0140625	0	0,423076923	0	1,7094E-05	3,4188E-06
Δ2			0	0,000560776				
δ2			0	0,000269604				

Задание 3. Описать в модуле функции, которые возвращают приближенные значения интегралов от функции f(x) с оценкой точности по принципу Рунге для методов центральных прямоугольников, трапеций и парабол.

Исходными данными являются: подынтегральная функция f(x); пределы интегрирования a, b; начальное число отрезков разбиения n; точность вычисления ε. Необходимые для работы значения подынтегральной функции вычисляются непосредственной подстановкой значений аргумента в вычислительную формулу функции.

Содержимое заголовочного файла integrals.h:

```
#ifndef CODE_INTEGRALS_H
#define CODE_INTEGRALS_H

#include "vector"


typedef double Function(double x);

double calculateIntegralByCentralRectangle(Function function, double lowBorder,
                                          double highBorder, int nParts);

double calculateIntegralByTrapezoid(Function function, double lowBorder,
                                   double highBorder, int nParts);

double calculateIntegralByParable(Function function, double lowBorder,
                                 double highBorder, int nParts);

double calculateIntegralByCentralRectangleWithEps(Function function, double lowBorder,
                                                 double highBorder, int nParts, double &eps);

double calculateIntegralByTrapezoidWithEps(Function function, double lowBorder,
                                          double highBorder, int nParts, double &eps);

double calculateIntegralByParableWithEps(Function function, double lowBorder,
                                        double highBorder, int nParts, double &eps);

struct gaussCoefficient {
  double t;
  double A;
};
```

```
double calculateIntegralByGauss(Function function, double lowBorder,
                        double highBorder, int nParts);

#endif //CODE_INTEGRALS_H
```

Содержимое файла реализации integrals.cpp:

```
#include "integrals.h"
#include "utility"
#include "vector"
#include "stdlib.h"


typedef std::pair<double, double> GridElement;

typedef std::vector<GridElement> Grid;

double calculateIntegralByCentralRectangle(Function function, double lowBorder,
                                double highBorder, int nParts) {
  Grid grid(nParts);

  double step = (highBorder - lowBorder) / nParts;
  double sum = 0;
  for (int i = 0; i < nParts; i++) {
    grid[i].first = lowBorder + (i + 0.5) * step;
    grid[i].second = function(grid[i].first);
    sum += grid[i].second;
  }

  return sum * step;
}

double calculateIntegralByTrapezoid(Function function, double lowBorder,
                          double highBorder, int nParts) {
  Grid grid(nParts + 1);

  grid[0] = {lowBorder, function(lowBorder)};
  grid[grid.size() - 1] = {highBorder, function(highBorder)};

  double step = (highBorder - lowBorder) / nParts;
  double sum = (grid[0].second + grid[grid.size() - 1].second) / 2;
  for (int i = 1; i < nParts; i++) {
    grid[i].first = lowBorder + i * step;
    grid[i].second = function(grid[i].first);
    sum += grid[i].second;
  }

  return step * sum;
}

double calculateIntegralByParable(Function function, double lowBorder,
                         double highBorder, int nParts) {
  Grid grid(nParts + 1);

  grid[0] = {lowBorder, function(lowBorder)};
  grid[grid.size() - 1] = {highBorder, function(highBorder)};
```

```
    double step = (highBorder - lowBorder) / nParts;
    double sum = grid[0].second + grid[grid.size() - 1].second;

    for (int i = 1; i < grid.size() - 1; i += 2) {
        grid[i].first = lowBorder + i * step;
        grid[i].second = function(grid[i].first);
        sum += 4 * grid[i].second;
    }

    for (int i = 2; i < grid.size() - 1; i += 2) {
        grid[i].first = lowBorder + i * step;
        grid[i].second = function(grid[i].first);
        sum += 2 * grid[i].second;
    }

    return step / 3 * sum;
}

double calculateIntegralByCentralRectangleWithEps(Function function, double lowBorder,
                                                  double highBorder, int nParts, double &eps) {
    double integralValueNParts = calculateIntegralByCentralRectangle(function, lowBorder, highBorder,
nParts);
    double difference = abs(integralValueNParts - calculateIntegralByCentralRectangle(function,
lowBorder,
                                                                   highBorder, nParts / 2));
    eps = (double) 1 / 3 * difference;
    return integralValueNParts;
}

double calculateIntegralByTrapezoidWithEps(Function function, double lowBorder,
                                           double highBorder, int nParts, double &eps) {
    double integralValueNParts = calculateIntegralByTrapezoid(function, lowBorder, highBorder,
nParts);
    double difference = abs(integralValueNParts - calculateIntegralByTrapezoid(function, lowBorder,
                                                                highBorder, nParts / 2));
    eps = (double) 1 / 3 * difference;
    return integralValueNParts;
}

double calculateIntegralByParableWithEps(Function function, double lowBorder,
                                         double highBorder, int nParts, double &eps) {
    double integralValueNParts = calculateIntegralByParable(function, lowBorder, highBorder, nParts);
    double difference = abs(integralValueNParts - calculateIntegralByParable(function, lowBorder,
                                                              highBorder, nParts / 2));
    eps = (double) 1 / 15 * difference;
    return integralValueNParts;
}
```

Задание 4. Составить программу для вычисления приближенных значений интегралов согласно варианту (все функции из таблицы вариантов заданий) с использованием всех функций, описанных в модуле.

Содержимое файла main.cpp:

```
#include <iostream>
#include "libs/integrals/integrals.h"
```

```
#include "cmath"
#include "windows.h"

double f1(double x) {
 return 6 * pow(x, 2) - 2 * x + 3;
}

double f2(double x) {
 return sin(x) * cos(x);
}

double f3(double x) {
 return 1 + log(2 * x);
}

int main() {
   SetConsoleOutputCP(CP_UTF8);

   std::cout << "Вычисление по формуле центральных прямоугольников\n" <<
        "Интеграл 1, n = 8: " << calculateIntegralByCentralRectangle(f1, -2, 1, 8) <<
        "\nИнтеграл 2, n = 8: " << calculateIntegralByCentralRectangle(f2, 0, M_PI / 3, 8)
<<
        "\nИнтеграл 3, n = 8: " << calculateIntegralByCentralRectangle(f3, 1, 2, 8) <<
        "\n\nВычисление по формуле трапеций\n" <<
        "Интеграл 1, n = 8: " << calculateIntegralByTrapezoid(f1, -2, 1, 8) <<
        "\nИнтеграл 2, n = 8: " << calculateIntegralByTrapezoid(f2, 0, M_PI / 3, 8) <<
        "\nИнтеграл 3, n = 8: " << calculateIntegralByTrapezoid(f3, 1, 2, 8) <<
        "\n\nВычисление по формуле парабол\n" <<
        "Интеграл 1, n = 8: " << calculateIntegralByParable(f1, -2, 1, 8) <<
        "\nИнтеграл 2, n = 8: " << calculateIntegralByParable(f2, 0, M_PI / 3, 8) <<
        "\nИнтеграл 3, n = 8: " << calculateIntegralByParable(f3, 1, 2, 8);
}
```

Результат работы программы:

```
D:\BGTU\VicMat\Lab3\Example\Code\cmake-build-debug\Code.exe
Вычисление по формуле центральных прямоугольников
Интеграл 1, n = 8: 29.7891
Интеграл 2, n = 8: 0.376073
Интеграл 3, n = 8: 2.07977

Вычисление по формуле трапеций
Интеграл 1, n = 8: 30.4219
Интеграл 2, n = 8: 0.372856
Интеграл 3, n = 8: 2.07879

Вычисление по формуле парабол
Интеграл 1, n = 8: 30
Интеграл 2, n = 8: 0.37501
Интеграл 3, n = 8: 2.07944
Process finished with exit code 0
```

Задание 5. Вычислить «вручную» интеграл из столбца 1 табл. 3.1 по формуле Гаусса при n = 2. Значения узлов ti и весов Ai приведены в табл. 3.2.

$$\int_{-2}^{1}(6x^2-2x+3)dx = \frac{1-(-2)}{2}\sum_{i=0}^{1} A_i f\left(\frac{1+(-2)}{2} + \frac{1-(-2)}{2}\cdot t_i\right) =$$

$$= \frac{3}{2}\left(6\left(\frac{3}{2}\cdot\left(-\frac{1}{\sqrt{3}}\right)-\frac{1}{2}\right)^2-2\left(\frac{3}{2}\cdot\left(-\frac{1}{\sqrt{3}}\right)-\frac{1}{2}\right)+3\right)+6\left(\frac{3}{2}\cdot\left(\frac{1}{\sqrt{3}}\right)-\frac{1}{2}\right)^2-2\left(\frac{3}{2}\cdot\left(\frac{1}{\sqrt{3}}\right)-\frac{1}{2}\right)+3\right)\Big) \approx 30{,}75$$

Задание 6. Описать в модуле функцию для вычисления приближенного значения интеграла от функции f(x) по формуле Гаусса при n = 1, 2, 3, 4. Добавить в составленную программу вычисление приближенных значений интегралов (все функции из таблицы вариантов заданий) с использованием формулы Гаусса.

Код программы:

```
struct gaussCoefficient {
 double t;
 double A;
};
std::vector<std::vector<gaussCoefficient>> gaussCoefficients{
  {},
  {{0, 2}},
  {{-0.57735, 1}, {0.57735, 1}},
  {{-0.77459, 0.55555}, {0, 0.88888}, {0.77459, 0.55555}},
  {{-0.86114, 0.34785}, {-0.33998, 0.65215}, {0.33998, 0.65215},
    {0.861136, 0.34785}}
};
double calculateIntegralByGauss(Function function, double lowBorder,
                  double highBorder, int nParts) {
 double sum = 0;
 for (int i = 0; i < nParts; i++) {
  double value = gaussCoefficients[nParts][i].A;
  value *= function((highBorder + lowBorder) / 2 +
              (highBorder - lowBorder) / 2 *
              gaussCoefficients[nParts][i].t);
  sum += value;
 }
 return (highBorder - lowBorder) / 2 * sum;
}

int main() {
 SetConsoleOutputCP(CP_UTF8);
 std::cout << "Вычисление интеграла по формуле Гаусса\n";
 for (int i = 1; i <= 4; i++) {
  std::cout << "Интеграл 1, n = " << i << ": " <<
        calculateIntegralByGauss(f1, -2, 1, i) << "\n";
 }
 std::cout << "Интеграл 2, n = 4: " <<
       calculateIntegralByGauss(f2, 0, M_PI / 3, 4) << "\n";
 std::cout << "Интеграл 3, n = 4: " <<
       calculateIntegralByGauss(f3, 1, 2, 4) << "\n";
}
```

```
D:\BGTU\VicMat\Lab3\Example\Code\cmake-build-debug\Code.exe
Вычисление интеграла по формуле Гаусса
Интеграл 1, n = 1: 16.5
Интеграл 1, n = 2: 30
Интеграл 1, n = 3: 29.9995
Интеграл 1, n = 4: 29.9999
Интеграл 2, n = 4: 0.375
Интеграл 3, n = 4: 2.07944

Process finished with exit code 0
```

Задание 7. Заполнить значения погрешности вычисления интеграла 1 в таблице 3.1 для формулы Гаусса при n = 1, 2, 3, 4.

Погрешности	Формула центральных прямоугольников		Формула трапеций	Формула парабол n = 8	Формула Гаусса			
	n = 8	n = 10	n = 8		n = 1	n = 2	n = 3	n = 4
Δ1	0,2109375	0,135	0,421875	0	12,375	0	0,0005	0,0001
δ1	0,00703125	0,0045	0,0140625	0	0,423076923	0	1,7094E-05	3,4188E-06
Δ2			0	0,000560776	**13,5**	**0**	**0,0005**	**0,0001**
δ2			0	0,000269604	**0,45**	**0**	**1,66667E-05**	**3,33333E-06**

Вывод: в ходе лабораторной работы мы изучили понятие квадратурной формулы; изучили основные способы численного интегрирования; получили практические навыки решения задачи численного интегрирования с помощью ЭВМ.