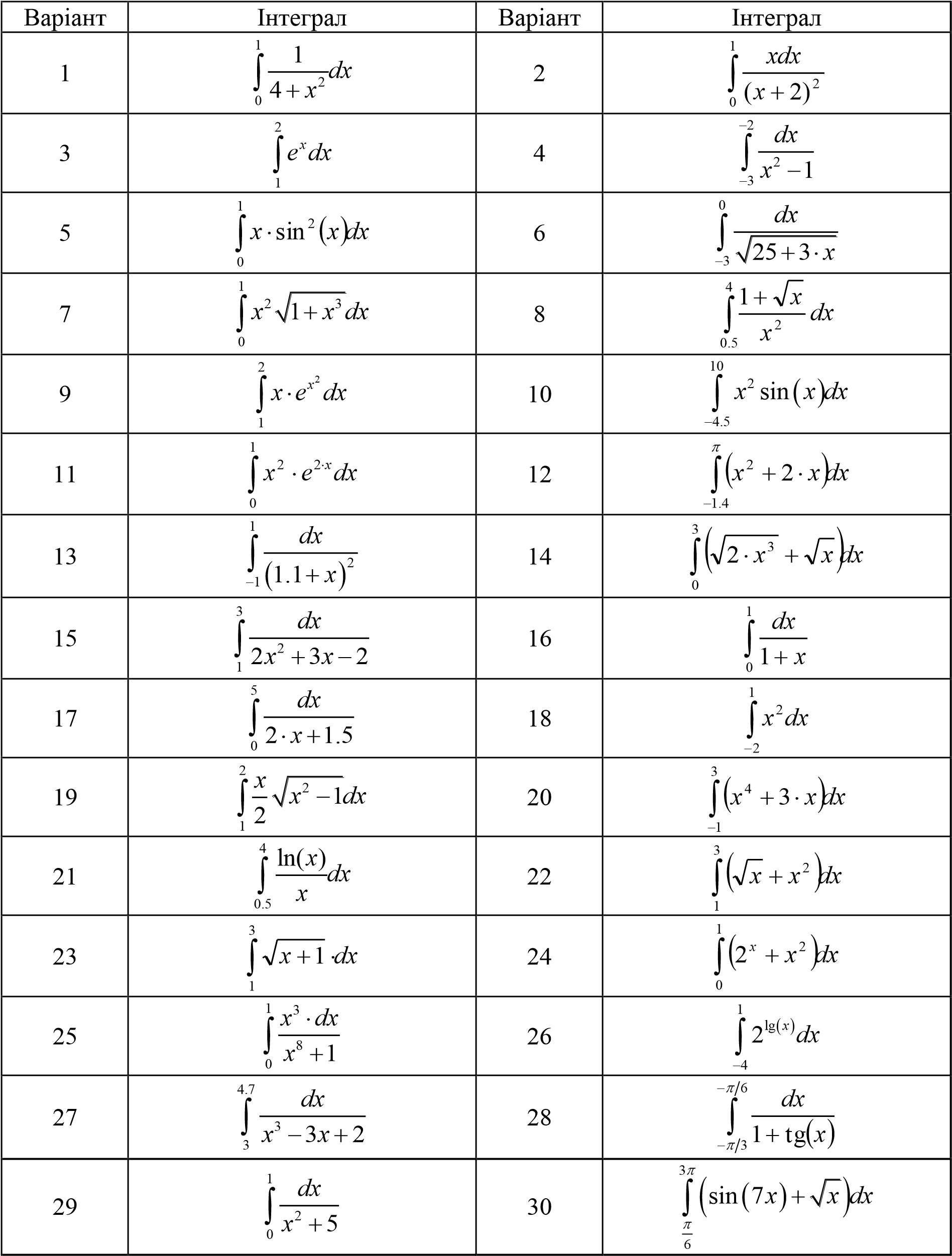
|  |  |
| --- | --- |
|  | Виконав: Бурлака Тимофій Сергійович  Група: РЕ-11  Викладачі: доцент Катін П.Ю.  Оцінка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Підпис: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Київ – 2021

**Мета роботи**: вивчити методи чисельного інтегрування; скласти програму обчислення визначеного інтегралу чисельними методами; дослідити залежність точності розрахунку інтегралу від кількості проміжків розбиття інтервалу інтегрування.

**Інтеграл**:



1)На початку програми я підключаю бібліотеку math.h, і записую 2 константи, це будуть межі мого інтегралу:

#include <math.h>

#define a 0.0

#define b 3.0

2)Вибираю яким метод я хочу отримати відповідь:

printf("Choise method: \n");

printf("1 - Method of rectangles\n2 - Method of trapezoidal\n3 - Simpson method\n4 - Usual method\n5 - My integral y = sqrt(x)\n");

scanf("%d",&var);

3)Для обрахунку кожного з методів я зробив окремі функції:

double Pramokytnik (int n) – Метод прямокутників

double Trap(int n)– Метод трапецій

double Simpson(int n) – Метод Сімпсона

4)Для виводу інформації на консоль також використовую функцію:

void Dump(double S,double Fault,int n )

**Код:**

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define a 0.0

#define b 1.0

void Dump(double S,double Fault,int n )

{

printf("S = %lf\n",S);

printf("fault = %lf\n",Fault);

printf("Count of steps = %d\n",n);

printf("ALL OK\n\n");

}

double Roz(double x)

{

return (x/(pow(x+2,2)));

}

double Pramokytnik(int n)

{

double h,S,x;

h=(b-a)/n;

for(x = a; x <= b ;x = x + h)

{

S=S+Roz(x);

}

S=S\*h;

return S;

}

double Trap(int n)

{

double h,S = 0,x;

h=(b-a)/n;

S=Roz(a)+Roz(b);

for(x = a; x <= b ;x = x + h)

{

S=S+Roz(x);

}

S=S\*h;

return S;

}

double Simpson(int n)

{

double h,S = 0,HS = 0;

int i = 0;

h=((b-a)/n);

for(i = 0; i <= n;i++)

{

if(i%2 != 0)

{

S=S+Roz(a+h\*i);

}

if(i%2 == 0)

{

HS=HS+Roz(a+h\*i);

}

}

S=((b-a)/(3\*n))\*(Roz(a)+4\*S+2\*HS+Roz(b));

return S;

}

int main()

{

/\*

MY VAR 2

integral ot 0 do 1 (x/(pow(x+2,2)))

\*/

int var,n,i,j=1;

double h,d,x,S=0.0,HS=0.0;

while(1)

{

printf("Choise method: \n");

printf("1 - Method of rectangles\n2 - Method of trapezoidal\n3 - Simpson method\n4 - Usual method\n5 - My integral y = sqrt(x)\n");

scanf("%d",&var);

switch("%d",var)

{

case 1://rectangles

printf("Count of steps: \n");

scanf("%d",&n);

for(j; j<=2;j++)

{

if(j==1)

{

S = Pramokytnik(n);

n=n+2;

}

HS = Pramokytnik(n);

if(fabs(S-HS)>0.00001 && fabs(S-HS)<0.001)

{

HS = fabs(S-HS);

Dump(S,HS,n);

break;

}

else

{

j=0;

}

}

break;

case 2://trapezoidal

printf("Count of steps: \n");

scanf("%d",&n);

for(j; j<=2;j++)

{

if(j==1)

{

S = Trap(n);

n=n+2;

}

HS = Trap(n);

if(fabs(S-HS)>0.00001 && fabs(S-HS)<0.001)

{

HS = fabs(S-HS);

Dump(S,HS,n);

break;

}

else

{

j=0;

}

}

break;

case 3://simpson

do

{

printf("This method must have an even number of intervals \n");

printf("Count of steps: \n");

scanf("%d",&n);

}

while(n%2!=0);

for(j; j<=2;j++)

{

if(j==1)

{

S = Simpson(n);

n=n+2;

}

HS = Simpson(n);

if(fabs(S-HS)>0.00001 && fabs(S-HS)<0.001)

{

HS = fabs(S-HS);

Dump(S,HS,n);

break;

}

else

{

j=0;

}

}

break;

case 4:

S=((2\*pow(a,2)\*sqrt(2\*a))/5)+((2\*a\*sqrt(a))/3);

HS=((2\*pow(b,2)\*sqrt(2\*b))/5)+((2\*b\*sqrt(b))/3);

S=HS-S;

printf("S : %lf\n\n",S);

break;

case 5:

//y = sqrt(x)

printf("My integral (0->1) y = sqrt(x)");

printf("Count of steps: \n");

scanf("%d",&n);

printf("Calculator answer : 0.6\n");

h = ((b-a)/n);

for(x = a; x <= b ;x = x + h)

{

S=S+sqrt(x);

}

S=S\*h;

printf("Method of rectangles : %lf\n",S);

//2

h=(b-a)/n;

S=sqrt(a)+sqrt(b);

for(x = a; x <= b ;x = x + h)

{

S=S+sqrt(x);

}

S=S\*h;

printf("Method of trapezoidal : %lf\n",S);

//3

h=((b-a)/n);

for(i = 0; i <= n;i++)

{

if(i%2 != 0)

{

S=S+sqrt(a+h\*i);

}

if(i%2 == 0)

{

HS=HS+sqrt(a+h\*i);

}

}

S=((b-a)/(3\*n))\*(sqrt(a)+4\*S+2\*HS+sqrt(b));

printf("Simpson method : %lf\n\n",S);

break;

default:

printf("No one variant\n");

break;

}

}

return 0;

}

