

**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук
Образовательная программа бакалавриата «Программная инженерия»

**Программа для интегрирования функции $y = a + b * x^{-2}$
в заданном диапазоне методом Симпсона**

Пояснительная записка

Исполнитель
студент группы БПИ198
Малышенко А. М.

Москва 2020

Содержание

1. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ.....	3
1.1 Постановка задачи на разработку программы.....	3
1.2 Описание алгоритма и функционирования программы.....	3
1.3 Описание функций.....	3
1.4 Формат входных данных.....	3
1.5 Формат выходных данных.....	3
ПРИЛОЖЕНИЕ 1.....	4
Приложение 2.....	9

1. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

1.1 Постановка задачи на разработку программы

Разработать программу для интегрирования функции $y = a + b * x^{-2}$ (задаётся двумя числами a,b) в заданном диапазоне(задается так же) методом Симпсона

1.2 Описание алгоритма и функционирования программы

Алгоритм реализует формулу метода Симпсона (парабол):

$$\int_a^b f(x) dx \approx \frac{h}{3} (f(x_0) + 4 \sum_{i=1}^n f(x_{2i-1}) + 2 \sum_{i=1}^{n-1} f(x_{2i}) + f(x_{2n})) ,$$

где $h = \frac{b-a}{2n}$, $n = 100$ - количество отрезков

Для облегчения вычислений $2n$ будет заменено на n и все вычисления будут использовать это правило.

Во время инициализации алгоритма считается константа h

Далее алгоритм сначала считает крайние элементы суммы, после чего в отдельных циклах считает частичные суммы для каждого слагаемого.

Полученную сумму программа множит на h и делит на 3. Результат выводится.

1.3 Описание функций

calculate_h: устанавливает $h = \frac{b-a}{n}$

add_result_to_sum: принимает **double result**, и обновляет переменную **sum += result**

calculate_2i_minus_1_cicrle: обновляет сумму, считая $4 \sum_{i=1}^n f(x_{2i-1})$

calculate_2i_cicrle: обновляет сумму, считая $2 \sum_{i=1}^{n-1} f(x_{2i})$

f: функция вычисления функции $f(x) = a + b * x^{-2}$. Принимает **double result** и возвращает результат вычислений(**double**)

1.4 Формат входных данных

В консоль вводятся последовательно, через пробел числа a, b, левая и правая граница. Левая граница должна быть меньше или равна правой границе.

1.5 Формат выходных данных

В консоль выводится результат вычисления интеграла или сообщение о некорректном вводе

ПРИЛОЖЕНИЕ 1

Исходный код программы:

```
; Малышенко Александр Михайлович
; 18 вариант
;
; Задача:
; Разработать программу интегрирования функции  $y=a+b*x^{-2}$  (задаётся двумя числами a,b)
; в заданном диапазоне(задаётся так же) методом Симпсона(использовать FPU)
;
; C++ equivalent solution
;
; int n = 200;
; double h;
; double a, b;
; double l, r;

; double f(double x) {
;     return a + b / (x * x);
; }

; int main() {
;     printf("Input a, b and left, right side via space: ");
;     scanf("%lf%lf%lf%lf", &a, &b, &l, &r);
;     if (l > r) {
;         printf("Left side mustn't be greater then right side");
;         return 0;
;     }
;     h = (r - l) / n;
;     double sum = f(l) + f(r);
;     for (int i = 1; i < n; i += 2) {
;         sum += 4 * f(l + h * i);
;     }
;     for (int i = 2; i < n; i += 2) {
;         sum += 2 * f(l + h * i);
;     }
;     printf("%lf", h * sum / 3);
;     return 0;
; }
```

format PE console

entry start

include 'win32a.inc'

section '.data' data readable writable

```
str_input_data db 'Input a, b and left, right side via space: ', 0
str_read_data db '%lf%lf%lf%lf', 0
str_result db 'Result is: %lf', 10, 0
str_left_side_greater_right_side db 'Left side mustn`'t be greater then right side', 0
str_d db '%lf', 0
```

```

n      dd 200
tmp     dd ?
h      dq ?
a      dq ?
b      dq ?
l      dq ?
r      dq ?
sum     dq 0
result  dq ?

```

section '.code' code readable executable

start:

```

invoke printf, str_input_data
invoke scanf, str_read_data, a, b, l, r

```

fini

```

; comp l and r

```

```

fld [r]

```

```

fld [l]

```

```

fcomi st, st1

```

```

ffree st0

```

```

ffree st1

```

```

jb l_less_or_equal_r

```

```

; l > r

```

```

invoke printf, str_left_side_greater_right_side

```

```

jmp finish

```

l_less_or_equal_r:

```

stdcall calculate_h ; h = (r - l) / n

```

```

fldz

```

```

fst [sum] ; sum = 0

```

```

ffree st0

```

```

fld [l]

```

```

stdcall f ; f(l)

```

```

stdcall add_result_to_sum ; sum += f(l)

```

```

fld [r]

```

```

stdcall f ; f(r)

```

```

stdcall add_result_to_sum ; sum += f(r)

```

```

; for (int i = 1; i < n; i += 2) {

```

```

;   sum += 4 * f(l + h * i);

```

```

; }

```

```

stdcall calculate_2i_minus_1_cicle

```

```

; for (int i = 2; i < n; i += 2) {

```

```

;   sum += 2 * f(l + h * i);

```

```
; }
stdcall calculate_2i_cicrle
```

```
fld [h]
fmul [sum]
mov [tmp], 3
fidiv [tmp]
fst [result] ; result = h * sum / 3
ffree st0
invoke printf, str_result, dword[result], dword[result + 4]
```

finish:

```
call [getch]
```

```
push 0
call [ExitProcess]
```

```
; calculate_h(): returns void
```

```
; sets h = (r - l) / n
```

```
calculate_h:
```

```
fld [r]
fsub [l]
fidiv [n]
fst [h] ; h = (r - l) / n
ffree st0
ret
```

```
; add_result_to_sum(double result): returns void
```

```
; updates sum += result
```

```
; result sends via FPU stack at st(0)
```

```
add_result_to_sum:
```

```
fld [sum]
fadd st, st1
fst [sum] ; sum += stack_top (result)
ffree st1
ffree st0
ret
```

```
; calculate_2i_minus_1_cicrle(): returns void
```

```
; does loop by following C++ instruction:
```

```
; for (int i = 1; i < n; i += 2) {
```

```
;   sum += 4 * f(1 + h * i);
```

```
; }
```

```
calculate_2i_minus_1_cicrle:
```

```
mov ebx, 1 ; i = 1
```

```
start_2i_minus_1_loop:
```

```
cmp ebx, [n]
```

```
jge end_2i_minus_1_loop
```

```
fld [h]
```

```
mov [tmp], ebx
```

```
fimul [tmp]
```

```
fadd [l] ; h*i + 1
```

```

    stdcall f    ; f(h*i + 1)
    mov [tmp], 4
    fimul [tmp] ; 4*f(h*i + 1)
    stdcall add_result_to_sum
    add ebx, 2
    jmp start_2i_minus_1_loop
end_2i_minus_1_loop:
    ret

```

```

; calculate_2i_cicrle(): returns void
; does loop by following C++ instruction:
; for (int i = 2; i < n; i += 2) {
;     sum += 2 * f(1 + h * i);
; }

```

```

calculate_2i_cicrle:

```

```

    mov ebx, 2; i = 2

```

```

start_2i_loop:

```

```

    cmp ebx, [n]

```

```

    jge end_2i_loop

```

```

    fld [h]

```

```

    mov [tmp], ebx

```

```

    fimul [tmp]

```

```

    fadd [1]    ; h*i + 1

```

```

    stdcall f    ; f(h*i + 1)

```

```

    mov [tmp], 2

```

```

    fimul [tmp] ; 2*f(h*i + 1)

```

```

    stdcall add_result_to_sum

```

```

    add ebx, 2

```

```

    jmp start_2i_loop

```

```

end_2i_loop:

```

```

    ret

```

```

; f(double x): returns double result

```

```

; calculates f(x) = a+b*x^-2

```

```

; x sends via FPU stack at st(0)

```

```

; result sends via FPU stack at st(0)

```

```

f:

```

```

    fld1

```

```

    fmul st0,st1

```

```

    fmul st,st1 ; x*x

```

```

    fld [b]

```

```

    fdiv st,st1 ; b / x*x

```

```

    fadd [a]    ; b / x*x + a

```

```

    ffree st1

```

```

    ffree st2

```

```

    ret

```

```

;-----third act - including HeapApi-----

```

```

section '.idata' import data readable

```

```

    library kernel, 'kernel32.dll',\

```

```

        msvcrt, 'msvcrt.dll',\

```

```

        user32, 'USER32.DLL'

```


```
include 'api\user32.inc'  
include 'api\kernel32.inc'  
    import kernel,\  
        ExitProcess, 'ExitProcess',\  
        HeapCreate, 'HeapCreate',\  
        HeapAlloc, 'HeapAlloc'  
include 'api\kernel32.inc'  
    import msvcrt,\  
        printf, 'printf',\  
        scanf, 'scanf',\  
        getch, '_getch'
```


Приложение 2

Тестирование программы:

Корректность работы проверяется с помощью онлайн калькулятора

Тест 1.

 C:\Users\Alex\Documents\testFolder\test.EXE

```
Input a, b and left, right side via space: 1 1 1 2
Result is: 1.500000
```

Представим исходный интеграл, как сумму интегралов:

$$\int 1 + x^{-2} dx = \int 1 dx + \int x^{-2} dx$$

$$\int 1 dx$$

Это табличный интеграл:

$$\int 1 dx = x + C$$

$$\int x^{-2} dx$$

Это табличный интеграл:

$$\int x^{-2} dx = -\frac{1}{x} + C$$

$$\int 1 + x^{-2} dx = x - \frac{1}{x} + C$$

Вычислим определенный интеграл:


$$\int_1^2 1 + x^{-2} dx = \left(x - \frac{1}{x} \right) \Big|_1^2$$

$$F(2) = \frac{3}{2}$$

$$F(1) = 0$$


$$I = \frac{3}{2} - (0) = \frac{3}{2}$$

Тест 2.

 C:\Users\Alex\Documents\testFolder\test.EXE

```
Input a, b and left, right side via space: 9 9 -2 -3
Left side mustn't be greater then right side
```

Тест 3.

 C:\Users\Alex\Documents\testFolder\test.EXE

```
Input a, b and left, right side via space: 123 333 1 1200
Result is: 148226.241499
```

Представим исходный интеграл, как сумму интегралов:

$$\int 123 + 333 \cdot x^{-2} dx = \int 123 dx + \int \frac{333}{x^2} dx$$

$$\int 123 dx$$

Это табличный интеграл:

$$\int 123 dx = 123 \cdot x + C$$

$$\int \frac{333}{x^2} dx$$

Это табличный интеграл:

$$\int \frac{333}{x^2} dx = -\frac{333}{x} + C$$

$$\int 123 + 333 \cdot x^{-2} dx = 123 \cdot x - \frac{333}{x} + C$$

Вычислим определенный интеграл:


$$\int_1^{1200} 123 + 333 \cdot x^{-2} dx = \left(123 \cdot x - \frac{333}{x} \right) \Big|_1^{1200}$$

$$F(1200) = \frac{59039889}{400}$$

$$F(1) = -210$$


$$I = \frac{59039889}{400} - (-210) = \frac{59123889}{400}$$

=147809,7225. n здесь уже мало, увеличиваем до n=20000

 C:\Users\Alex\Documents\testFolder\test.EXE

```
Input a, b and left, right side via space: 123 333 1 1200
Result is: 147809.723066
```

Тест 4.

 C:\Users\Alex\Documents\testFolder\test.EXE

```
Input a, b and left, right side via space: -1000 -1000 -1000 -1
Result is: -999999.000823
```

Представим исходный интеграл, как сумму интегралов:

$$\int -1000 - 1000 \cdot x^{-2} dx = \int -1000 dx + \int -\frac{1000}{x^2} dx$$

$$\int (-1000) dx$$

Это табличный интеграл:

$$\int -1000 dx = -1000 \cdot x + C$$

$$\int \left(-\frac{1000}{x^2}\right) dx$$

Это табличный интеграл:

$$\int -\frac{1000}{x^2} dx = \frac{1000}{x} + C$$

$$\int -1000 - 1000 \cdot x^{-2} dx = -1000 \cdot x + \frac{1000}{x} + C$$

Вычислим определенный интеграл:

$$\int_{-1000}^{-1} -1000 - 1000 \cdot x^{-2} dx = \left(-1000 \cdot x + \frac{1000}{x}\right) \Big|_{-1000}^{-1}$$

$$F(-1) = 0$$

$$F(-1000) = 999999$$

$$I = 0 - (999999) = -999999$$