Подп. и дата	
Инв. № дубл.	
Взам. инв. №	
Подп. и дата	
Инв. № подл	

ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Образовательная программа бакалавриата «Программная инженерия»

Исполнитель: студент группы БПИ197(1) Галкин Никита Сергеевич

Микропроект №1

Пояснительная записка

Москва 2020 г.

Подп. и дата	
Инв. № дубл.	
Взам. инв. №	
Подп. и дата	
Инв. № подл	

Микропроект №1

Листов 12

Москва 2020 г.

Содержание

1. ФОРМУЛИРОВКА ЗАДАНИЯ

Разработать программу численного интегрирования функции y=a+b*x3 (задаётся целыми числами a,b) в определённом диапазоне целых (задаётся так же) методом прямоугольников с недостатком (шаг 1)

2. ПРОЦЕДУРЫ ПРОГРАММЫ

Таблица 2.1

ПРОЦЕДУРЫ	Назначение
start	Начало программы.
zeroAlgo	Алгоритм, вызываемый при равности введенных x.
normalAlgo	Нормальный алгоритм, вызываемый при x1 < x2.
negativeAlgo	Алгоритм, вызываемый при обратном порядке x'ов (x1 > x2).
printResult	Вывод ненулевого результата.
printResultZero	Вывод нулевого результата (если x1 = x2)
mainCycle	Цикл нахождения интеграла.
inverse	Меняет знак ebx (интеграла) на противоположный.
continue	Алгоритм продолжения цикла.
finish	Алгоритм завершения работы программы.

3. ПРИМЕНЯЕМЫЕ РАСЧЕТНЫЕ МЕТОДЫ

Для реализации нахождения определенного интеграла функции используется метод правых прямоугольников (рис. 1).

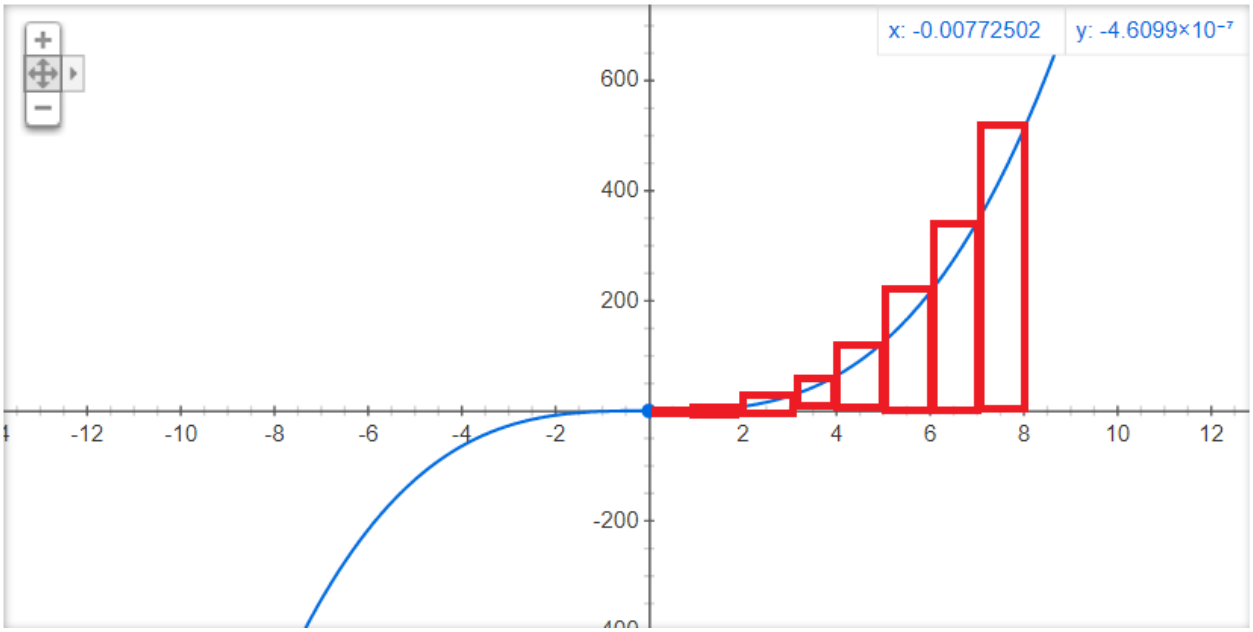

Рисунок 1. Метод правых прямоугольников

Смотрится, если x1 меньше или равен x2, иначе они меняются местами. Программа проходит циклом от x1 до x2 с шагом 1 (можно выставить в программе), находит значение ординаты для правого значения абсциссы (для каждого из таких прямоугольников), находит площадь каждого прямоугольника (шаг*y, может быть отрицательной) и складывает их. Если x1 изначально был меньше x2, то эта сумма и есть искомый интеграл. В обратном случае необходимо поменять знак интеграла на противоположный. В результате выйдет интеграл функции y = a + b*(x**3) на отрезке [x1, x2].

4. ТЕСТИРОВАНИЕ ПРОГРАММЫ

При запуске программы открывается консоль, выводится информация о формуле функции, также выведется текстовый запрос на ввод «a» (рис. 2).

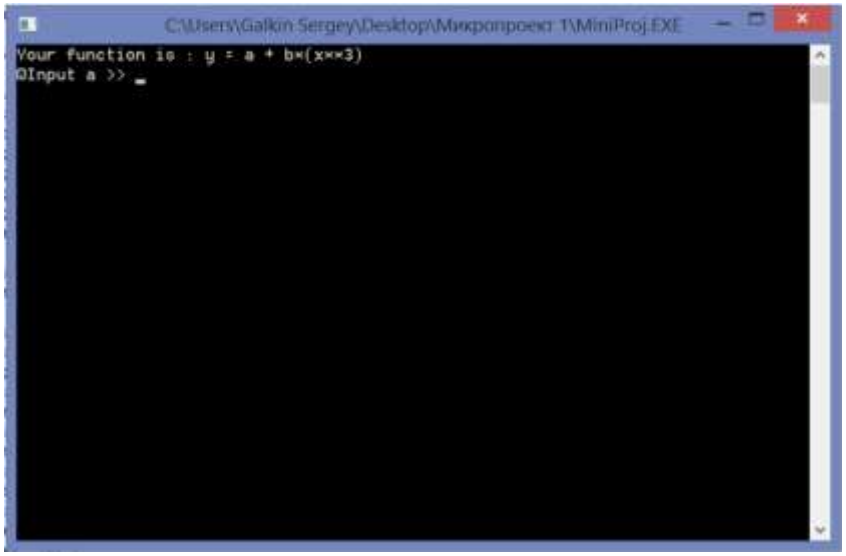

Рисунок 2. Запуск программы

Далее необходимо вписать значение «a» и нажать Enter. После этого, аналогично запросу на «a», будут проведены запросы на «b», «x1», «x2». После ввода «x2» программа подсчитает ответ и выведет его в консоль (рис. 3).

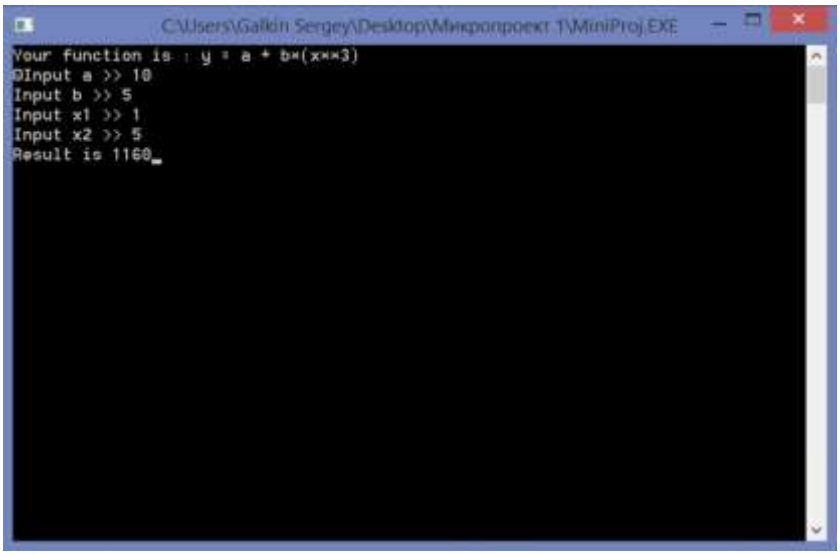

Рисунок 3. Пример работы программы

На значения параметров функции («a» и «b») и концов отрезка интегрирования («x1» и «x2») есть ограничение: необходимо, чтобы они были целыми числами.

ПРИЛОЖЕНИЕ №1

ТЕКСТ ПРОГРАММЫ

```
; Галкин Никита Сергеевич, БПИ197, Микропроект №1.
format PE console
entry start

include 'win32a.inc'

section '.data' data readable writable
         ; Текстовые формы.
         displayY db "Your function is : y = a + b*(x**3)", 0
         getAInfo db "Input a >> ", 0
         getBInfo db "Input b >> ", 0
         getx1Info db "Input x1 >> ", 0
         getx2Info db "Input x2 >> ", 0
         resultInfo db "Result is %d", 0

         ; Форма под ввод данных.
         spaceStr db "%d", 0

         ; Ячейки памяти.
         ; Перенос на другую строку.
         gap db 0x0A
         ; Шаг.
         step dd 1
         ; a из формулы.
         a   dd ?
         ; b из формулы.
         b   dd ?
         ; Отрезок интегрирования.
         x1  dd ?
         x2  dd ?

section '.code' code readable executable
; Начало программы.
start:
        ; Вывод формулы функции.
        push displayY
        call [printf]

        ; Перенос на другую строку после формулы функции.
        push gap
        call [printf]

        ; Текстовый запрос на ввод a.
        push getAInfo
        call [printf]

        ; Получение a.
        push a
        push spaceStr
        call [scanf]

        ; Текстовый запрос на ввод b.
        push getBInfo
        call [printf]

        ; Получение b.
        push b
```

```
        push spaceStr
        call [scanf]

        ; Текстовый запрос на ввод x1.
        push getx1Info
        call [printf]

        ; Получение x1.
        push x1
        push spaceStr
        call [scanf]

        ; Текстовый запрос на ввод x2.
        push getx2Info
        call [printf]

        ; Получение x2.
        push x2
        push spaceStr
        call [scanf]

        mov edx, [x1]       ; Для сравнения x1 и x2 помещаю значение x1 в
регистр edx.

        cmp edx, [x2]       ; если x1 < x2
        jl normalAlgo

        cmp edx, [x2]       ; если x1 = x2
        je zeroAlgo

        call negativeAlgo   ; если x1 > x2

; Алгоритм, вызываемый при обратном порядке x'ов (x1 > x2).
negativeAlgo:
        ; Обмен x местами.
        mov edx, [x1]
        xchg edx, [x2]
        mov [x1], edx
        ; Сбор первичного значения x в регистр edx.
        mov edx, [x1]
        add edx, [step]
        mov eax, edx        ; Перенос первичного значения x в eax.
        mov ebx, 0          ; Сумма площадей (в результате - интеграл).
        call mainCycle      ; Получение интеграла.

        call inverse        ; Для обратного порядка иксов необходимо
поменять знак результата.

; Алгоритм, вызываемый при равности введенных x.
zeroAlgo:
        call printResultZero ; Вывод нулевого результата.

; Нормальный алгоритм, вызываемый при x1 < x2.
normalAlgo:
        ; Сбор первичного значения x в регистр edx.
        mov edx, [x1]
        add edx, [step]
        mov eax, edx        ; Перенос первичного значения x в eax.
        mov ebx, 0          ; Сумма площадей (в результате - интеграл).
        call mainCycle      ; Получение интеграла.
```

```
        call printResult   ; Выводим результат.

; Вывод результата в консоль.
printResult:
        push ebx
        push resultInfo
        call [printf]

        call finish

; Вывод нулевого результата (если x1 = x2)
printResultZero:
        push 0
        push resultInfo
        call [printf]
        call finish

; Цикл нахождения интеграла.
mainCycle:
        mov ecx, eax ; Сохранение значения eax (смысл в хранении текущей
позиции).

        ; Подсчет y по формуле функции (собирается в eax).
        mov edx, eax
        imul eax, edx
        imul eax, edx      ; Возведение в куб.
        imul eax, [b]      ; Умножение на b.
        add eax, [a]       ; Сложение с a.
        mul [step]         ; Получение площади участка.

        add ebx, eax       ; Добавление площади к счетчику (собираемый
интеграл).

        mov eax, ecx       ; Возвращение старого значения eax (текущая
позиция).

        cmp eax, [x2]      ; Проверка на окончание цикла.
        jne continue

        ret

; Меняет знак ebx (интеграла) на противоположный.
inverse:
        mov eax, ebx       ; Сохраняем интеграл в регистре eax.

        cdq                ; Увеличиваем размерность регистра eax.
        imul eax, -1       ; Меняем знак интеграла.

        mov ebx, eax       ; Возвращаем значение интеграла в ebx.

        call printResult   ; Выводим результат.

; Алгоритм продолжения цикла.
continue:
        add eax, [step]    ; Увеличение x на шаг.
        jmp mainCycle      ; Повторный цикл.

; Алгоритм завершения работы программы.
finish:
        call [getch]
        push 0
```

```
        call [ExitProcess]


;--------------------------------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

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1) Информация с сайта http://www.softcraft.ru/
2) Видеоуроки на платформе YouTube.