

**НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ УКРАИНЫ
“КИЕВСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ ИМЕНИ ИГОРЯ
СИКОРСКОГО”**

**ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ
КАФЕДРА ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ**

КОМПЬЮТЕРНЫЙ ПРАКТИКУМ №1

Дисциплина: «Методы вычислений»

Тема: «Решение нелинейных уравнений»

Выполнила

студентка 3 курса группы ФИ-41

Лавягина Ольга Алексеевна

Проверил

Стёпочкина Ирина Валерьевна

1 ИСХОДНЫЕ ДАННЫЕ

В компьютерном практикуме (вариант 8) ищутся корни уравнения

$$2x^5 + 3x^2 - 2x - 6 = 0. \quad (1.1)$$

2 ДОПРОГРАММНЫЙ ЭТАП

Корни уравнения были найдены с помощью WolframAlpha, построен график (рис. 2.1).

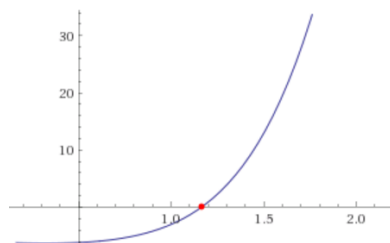


Рисунок 2.1 — График полинома $f(x) = 2x^5 + 3x^2 - 2x - 6$

Красной точкой на графике отмечен корень уравнения 1.1. Он лежит на промежутке от 1 до 1.5.

3 ЛИСТИНГ ПРОГРАММЫ

Листинг файла utils.py

```
a = 1
b = 1.5
accuracy = 10**(-5)

def f(x):
    return 2 * x**5 + 3 * x**2 - 2*x - 6

def output(iteration, x, accuracy):
    print 'Iteration # {}'.format(iteration)
    print 'Approximate value {}'.format(x)
    print 'Error: {}'.format(accuracy)
```

Листинг программы уточнения корней по методу бисекции

```
from utils import *

def bisection(a, b, accuracy):
    c = (a + b) / 2
    iteration = 1
    while abs(a - b) > accuracy:
        output(iteration, c, abs(a - b))
        if f(c) == 0:
            return c
        elif f(a) * f(c) < 0:
            b = c
        elif f(b) * f(c) < 0:
            a = c
        c = (a + b) / 2
        iteration += 1
    return c

bisection(a, b, accuracy)
```

Листинг программы уточнения корней по методу хорд

```
from utils import *

def horde(a, b, accuracy):
    c = (a * f(b) - b * f(a)) / (f(b) - f(a))
    iteration = 1
    while abs(f(c)) >= accuracy:
        output(iteration, c, abs(f(c)))
        if f(c) == 0:
            return c
        elif f(a) * f(c) < 0:
```

```

        b = c
    elif f(b) * f(c) < 0:
        a = c
    c = (a + b) / 2
    iteration += 1
return c

```

```
horde(a, b, accuracy)
```

Листинг программы уточнения корней по методу Ньютона (касательных)

```

from utils import *

def derivativeF(x):
    return 10 * x**4 + 6 * x - 2

def newton(x0, accuracy):
    iteration = 1
    while abs(f(x0)) >= accuracy:
        output(iteration, x0, abs(f(x0)))
        x0 = x0 - f(x0) / derivativeF(x0)
        iteration += 1
    return x0

newton(b, accuracy)

```

4 РЕЗУЛЬТАТЫ РАБОТЫ ПРОГРАММЫ

Результаты метода бисекции

```
Iteration # 1
Approximate value 1.25
Error: 0.5
Iteration # 2
Approximate value 1.125
Error: 0.25
Iteration # 3
Approximate value 1.1875
Error: 0.125
Iteration # 4
Approximate value 1.15625
Error: 0.0625
Iteration # 5
Approximate value 1.171875
Error: 0.03125
Iteration # 6
Approximate value 1.1640625
Error: 0.015625
Iteration # 7
Approximate value 1.16015625
Error: 0.0078125
Iteration # 8
Approximate value 1.162109375
Error: 0.00390625
Iteration # 9
Approximate value 1.1630859375
Error: 0.001953125
Iteration # 10
Approximate value 1.16357421875
Error: 0.0009765625
Iteration # 11
Approximate value 1.16333007812
Error: 0.00048828125
Iteration # 12
Approximate value 1.16345214844
Error: 0.000244140625
Iteration # 13
Approximate value 1.16351318359
Error: 0.0001220703125
Iteration # 14
Approximate value 1.16354370117
Error: 6.103515625e-05
Iteration # 15
Approximate value 1.16355895996
```

Error: 3.0517578125e-05
Iteration # 16
Approximate value 1.16355133057
Error: 1.52587890625e-05

Результаты метода хорд

Iteration # 1
Approximate value 1.09411764706
Error: 1.461142352
Iteration # 2
Approximate value 1.29705882353
Error: 3.79520343167
Iteration # 3
Approximate value 1.19558823529
Error: 0.782945057463
Iteration # 4
Approximate value 1.14485294118
Error: 0.424146888749
Iteration # 5
Approximate value 1.17022058824
Error: 0.156838704004
Iteration # 6
Approximate value 1.15753676471
Error: 0.139127423225
Iteration # 7
Approximate value 1.16387867647
Error: 0.00746674517207
Iteration # 8
Approximate value 1.16070772059
Error: 0.0661749749345
Iteration # 9
Approximate value 1.16229319853
Error: 0.029440596243
Iteration # 10
Approximate value 1.1630859375
Error: 0.0110085862703
Iteration # 11
Approximate value 1.16348230699
Error: 0.00177634078818
Iteration # 12
Approximate value 1.16368049173
Error: 0.00284384649989
Iteration # 13
Approximate value 1.16358139936
Error: 0.00053341401189
Iteration # 14
Approximate value 1.16353185317
Error: 0.000621548089255

```
Iteration # 15
Approximate value 1.16355662626
Error: 4.40882151951e-05
Iteration # 16
Approximate value 1.16356901281
Error: 0.000244657604066
Iteration # 17
Approximate value 1.16356281954
Error: 0.000100283370882
Iteration # 18
Approximate value 1.1635597229
Error: 2.80972469593e-05
```

Результаты метода Ньютона (касательных)

```
Iteration # 1
Approximate value 1.5
Error: 12.9375
Iteration # 2
Approximate value 1.27548806941
Error: 3.08131561768
Iteration # 3
Approximate value 1.17955664636
Error: 0.381874681655
Iteration # 4
Approximate value 1.1639290992
Error: 0.0086433492071
```


ВЫВОДЫ