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

Киев — 2017 года

1 ИСХОДНЫЕ ДАННЫЕ

В компьютерном практикуме (вариант 8) ищутся корни уравнения

$$2x^5 + 3x^2 - 2x - 6 = 0. \tag{1.1}$$

2 ДОПРОГРАММНЫЙ ЭТАП

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

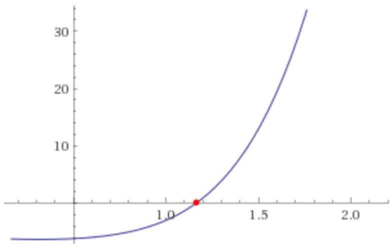

Рисунок 2.1 — График полинома $f\left(x\right) = 2x^5 + 3x^2 - 2x - 6$

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 True:
        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_prev, c = c, (a * f(b) - b * f(a)) / (f(b) - f(a))
        if abs(c_prev - c) < accuracy:
            break
        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 с критерием завершения процесса $|f\,(c)| < \varepsilon$

```
Iteration # 1
Approximate value 1.09411764706
Error: 1.461142352
Iteration # 2
Approximate value 1.13530568974
Error: 0.631659938106
Iteration # 3
Approximate value 1.15228262763
Error: 0.258503016609
Iteration # 4
Approximate value 1.15909423487
Error: 0.103381367024
Iteration # 5
Approximate value 1.16179675957
Error: 0.0409612978165
Iteration # 6
Approximate value 1.1628641622
Error: 0.0161694549738
Iteration # 7
Approximate value 1.16328499297
Error: 0.00637353689004
Iteration # 8
Approximate value 1.16345079075
Error: 0.00251081402306
Iteration # 9
Approximate value 1.16351609305
Error: 0.000988893788602
Iteration # 10
Approximate value 1.16354181065
Error: 0.000389444685057
Iteration # 11
Approximate value 1.16355193841
Error: 0.000153365108101
Iteration # 12
Approximate value 1.16355592672
Error: 6.03950443683e-05
Iteration # 13
Approximate value 1.16355749731
Error: 2.37833848757e-05
```

Результаты метода хорд с критерием завершения процесса

$$|c - c_{previous}| < \varepsilon$$

```
Iteration # 1
Approximate value 1.09411764706
Error: 1.461142352
Iteration # 2
Approximate value 1.13530568974
Error: 0.631659938106
Iteration # 3
Approximate value 1.15228262763
Error: 0.258503016609
Iteration # 4
Approximate value 1.15909423487
Error: 0.103381367024
Iteration # 5
Approximate value 1.16179675957
Error: 0.0409612978165
Iteration # 6
Approximate value 1.1628641622
Error: 0.0161694549738
Iteration # 7
Approximate value 1.16328499297
Error: 0.00637353689004
Iteration # 8
Approximate value 1.16345079075
Error: 0.00251081402306
Iteration # 9
Approximate value 1.16351609305
Error: 0.000988893788602
Iteration # 10
Approximate value 1.16354181065
Error: 0.000389444685057
Iteration # 11
Approximate value 1.16355193841
Error: 0.000153365108101
```

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

С помощью метода бисекции корень заданного уравнения был получен на 16-й итерации, с помощью метода хорд — на 13-й, а с помощью метода Ньютона (касательных) —на четвёртой.

Для метода бисекции было обнаружено, что при увеличении длины отрезка количество итераций возрастает, но при уменьшении длины отрезка практически не меняется. Так, для отрезка $[1.1, 1.2]$ результат был получен на 14-й итерации.

Метод хорд даёт результат гораздо быстрее. Для отрезка $[1.1, 1.2]$ результат был получен за на четвёртой итерации.

Преимуществом метода Ньютона (касательных) является его быстрая сходимость. Для данного метода необходимо знать начальное приближение, а не границы интервала, в котором находится корень. Недостатком является то, что метод может зацикливаться (в данном случае, например, при $x_0 = 1$).