МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ

Отчёт о лабораторной работе №3 по дисциплине основы программной инженерии

Выполнила:
Первых Дарья Александровна,
3 курс, группа ПИЖ-б-о-20-1

Проверил:
Доцент кафедры инфокоммуникаций,
Воронкин Р.А.

Ставрополь, 2023 г.

ВЫПОЛНЕНИЕ

Вычисление производных ¶

```
x = symbols('x')
diff(log(x), x, 3)
```

$$\frac{2}{x^3}$$

```
x = symbols('x')
y = x*cos(x)
diff(x*cos(x), x)
```

$$-x\sin(x)+\cos(x)$$

```
y = log(x**3,10)**3
diff(y,x,2).subs(x,10)
```

$$-\frac{9(-6+\log(1000))\log(1000)}{100\log(10)^3}$$

```
y = log(x**3,10)**3
diff(y,x,2).subs(x,10).simplify()
```

$$\frac{81(2-\log(10))}{100\log(10)^2}$$

```
y = (x**2+x-6)/(x**2-10*x+25)
z = diff(y,x)
solve(z, x)
```

```
[7/11]
```

Рисунок 1 – Пример выполнения программы

Производная функции, заданной в параметрической форме

```
t = symbols(' t')
x = t - sin(t)
y = 1 - cos(t)
y_diff = diff(y,t)/diff(x,t)
y_2diff = diff(y_diff,t)/diff(x,t)
y_2diff.simplify()
```

$$-\frac{1}{(\cos(t)-1)^2}$$

Односторонняя производная

```
import numpy as np
import matplotlib.pyplot as plt


%matplotlib inline
x = np.linspace(-2, 2, 500)
x[(x > -0.01) & (x < 0.01)] = np.nan
y = np.arctan(1 / x)
plt.plot(x, y)
plt.vlines(0, -1.6, 1.6, color = 'g', linestyles = 'dashed')
plt.show()
```

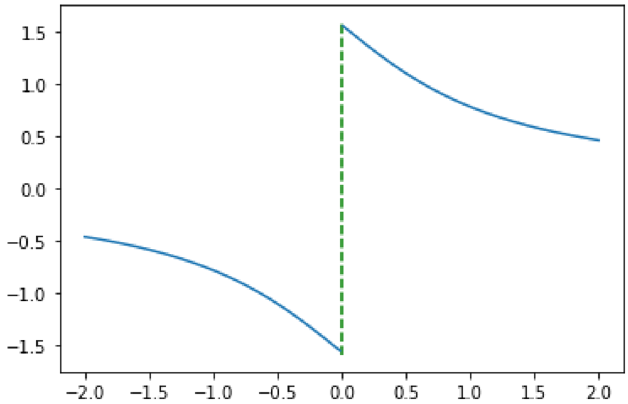


Рисунок 2 – Пример выполнения программы

```
y = atan(1/x)
z = diff(y,x)
limit(z, x, 0, dir="-")
```

$$0$$

```
limit(z, x, 0, dir="+")
```

$$0$$

```
t = symbols('t')
x = t - sin(t)
y = 1 - cos(t)
y_diff = diff(y, t) / diff(x, t)
y_diff
```

$$\frac{\sin(t)}{1 - \cos(t)}$$

```
y_2diff = diff(y_diff, t) / diff(x, t)
y_2diff.simplify()
```

$$-\frac{1}{(\cos(t) - 1)^2}$$

Рисунок 3 – Пример выполнения программы

Уравнение касательной ¶

```
x = np.linspace(-3, 4, 50)
y1 = x**2
plt.plot(x, y1, lw = 2, c = 'b')

x = np.linspace(-1, 4, 50)
y2 = 4*x - 4
plt.plot(x, y2, c = 'r')

plt.xlabel('x')
plt.ylabel('y')
plt.grid(True, linestyle = '-', color = '0.4')

plt.show()
```

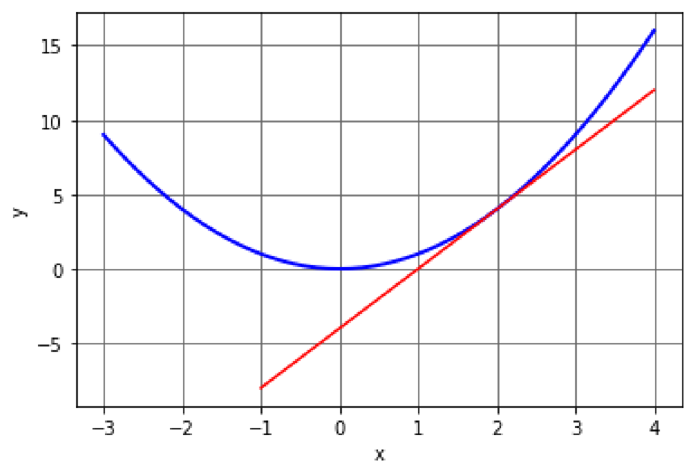
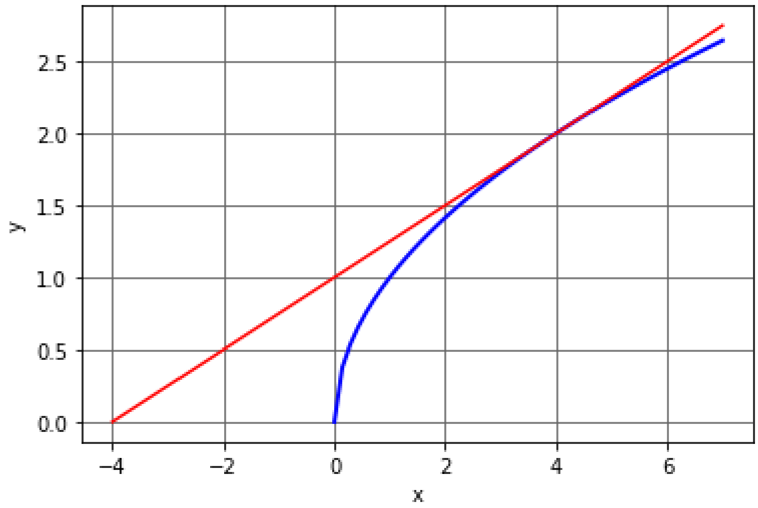


Рисунок 4 – Пример выполнения программы

```
x = np.linspace(0,7,50)
yl = np.sqrt(x)
pit.plot(x,yl,lw=2,c='b')
x = np.linspace(-4,7,50)
y2 = x/4 + 1
pit.plot(x,y2,c='r')
plt.xlabel('x')
pit.ylabel('y')
pit .grid (True, linestyle='-', color='0.4')
plt.show()
```




```
x = symbols('x')
y = 6*x**(1/3) + 2*sqrt(x)
x0 = 64
y0 = y.subs(x, x0)
l = tangent(y, 64)
l.equation()
```

$$-\frac{x}{4} + y - 24$$

```
p = Point(x0,y0)
l.perpendicular_line(p).equation()
```

$$-x - \frac{y}{4} + 74$$

Рисунок 5 – Пример выполнения программы

```
x = np.linspace(0, 120, 50)
y1 = 6*x**(1/3) + 2*x**(1/2)
plt.plot(x, y1, lw=2, c='r')

x = np.linspace(10, 120, 50)
y2 = x/4 + 24
plt.plot(x, y2, '--', lw=2, c='b')

x = np.linspace(60, 70, 50)
y3 = 296 - 4*x
plt.plot(x, y3, '-.', lw=2, c='g')

plt.xlabel('x')
plt.ylabel('y')
plt.grid(True, linestyle='-', color='0.4')
plt.axis('equal')
plt.show()
```

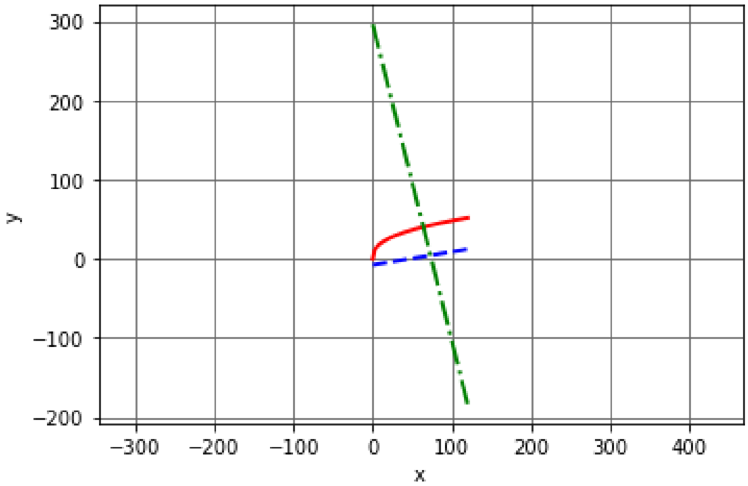


Рисунок 6 – Пример выполнения программы

```
def tangent_from_point(y, x1, y1):
    x, x0, y0 = symbols('x x0 y0')
    y_diff = diff(y, x).subs(x, x0)
    y_tang = y_diff*(x - x0) + y0
    first_eq = y.subs(x, x0) - y0
    second_eq = y_tang.subs(x, x1) - y1
    res = solve([first_eq, second_eq], \
                [x0, y0], dict = True)
    if len(res) == 1:
        x01 = res[0][x0]
        y01 = res[0][y0]
        return Line((x01, y01), (x1, y1))
    else:
        x021 = res[0][x0]; y021 = res[0][y0]
        x022 = res[1][x0]; y022 = res[1][y0]
        return Line((x021, y021), (x1, y1)), \
               Line((x022, y022), (x1, y1))
```

```
x = np.linspace(0, 7, 50)
y1 = np.sqrt(x)
plt.plot(x, y1, lw = 2, c = 'b')

x = np.linspace(-4, 7, 50)
y2 = x/4 + 1
plt.plot(x, y2, c = 'r')

plt.xlabel('x')
plt.ylabel('y')
plt.grid(True, linestyle = '-', color = '0.4')
plt.show()
```

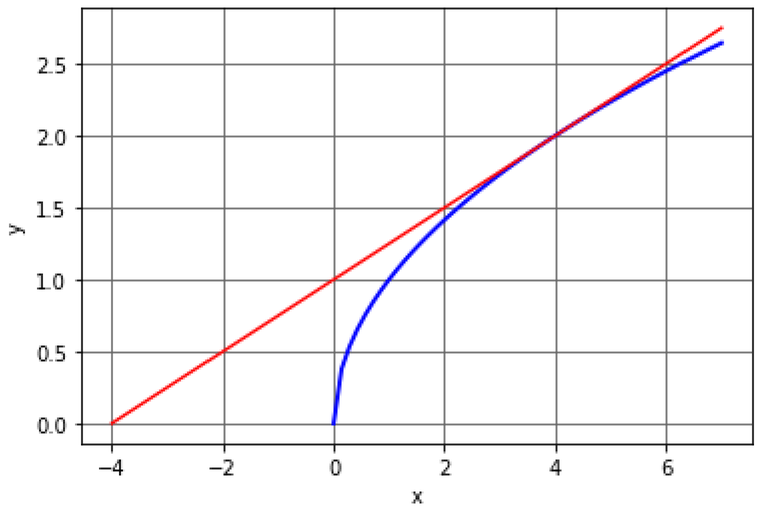


Рисунок 7 – Пример выполнения программы

Исследование функции

```
t = np.linspace(-1.5, 5, 100)
f = t**3 - 5*t**2 + 5
fd = 3*t**2 - 10*t
fdd = 6*t - 10
plt.plot(t, f, lw = 2, color = 'red', label = "$y(x)$")
plt.plot(t, fd, '--', lw = 2, color = 'b', label = "$y^{'}(x)$")
plt.plot(t, fdd, '-.', color = 'g', label = "$y^{''}(x)$")
plt.plot([0], [0], 'o', color = 'y')
plt.plot([0], [5], 'o', color = 'y')
plt.plot([3.3], [0], 'o', color = 'y')
plt.plot([3.3], [-13.4], 'o', color = 'y')
plt.plot([1.65], [0], 'o', color = 'b')
plt.plot([1.65], [-4], 'o', color = 'b')
plt.grid(True, linestyle = '-', color = '0.4')
plt.legend()
plt.show()
```

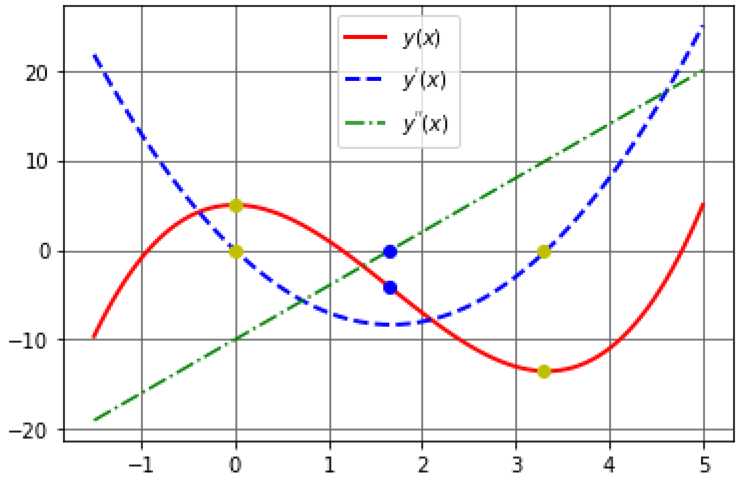


Рисунок 8 – Пример выполнения программы

Экстремум функции одной переменной

```
from scipy.optimize import minimize
x, y = symbols('x, y')
solve(x**2 < 3)
```

$$-\sqrt{3} < x \wedge x < \sqrt{3}$$

```
solve(x**2 - y**2, x)
```

```
[-y, y]
```

```
f = lambda x: np.exp(-x) - np.exp(-2*x)
```

```
x = np.linspace(0.1, 4, 50)
plt.plot(x, f(x), 'r')
plt.xlabel('X')
plt.ylabel('Y')
plt.grid(True, linestyle = '-', color = '0.4')
plt.show()
```

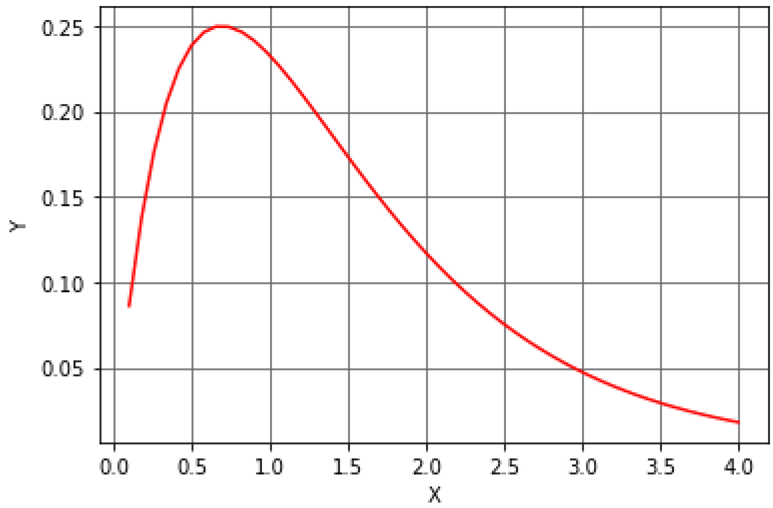


Рисунок 9 – Пример выполнения программы

```python
f_max = lambda x: -(np.exp(-x) - np.exp(-2*x))
res = minimize(f_max, -2)
print('x_max: %.3f f_max: %.3f' % (res.x, f(res.x)))
```

```
x_max: 0.693 f_max: 0.250
```

```python
res
```

```
      fun: -0.24999999999945666
 hess_inv: array([[1.98553383]])
      jac: array([-7.26431608e-07])
  message: 'Optimization terminated successfully.'
     nfev: 26
      nit: 12
     njev: 13
   status: 0
  success: True
        x: array([0.69314571])
```

```python
def tangent_plane(F, M):
    F_diff_x = diff(F, x).subs({x:M.x, y:M.y, z:M.z})
    F_diff_y = diff(F, y).subs({x:M.x, y:M.y, z:M.z})
    F_diff_z = diff(F, z).subs({x:M.x, y:M.y, z:M.z})
    n = Point(F_diff_x, F_diff_y, F_diff_z)
    p = Plane(M, normal_vector = n).equation()
    K = Point(M.x+n.x, M.y+n.y, M.z+n.z)
    l_n = Line(M, K).arbitrary_point()

    return p, l_n
```

```python
x, y, z = symbols('x y z')
M = Point(1, 1, 1)
F = x**2 + y**2 + z**2 - 9
p, l_n = tangent_plane(F, M)
```

```python
p
```

$$2x + 2y + 2z - 6$$

```python
l_n
```

$$Point3D(2t + 1, 2t + 1, 2t + 1)$$

Рисунок 10 – Пример выполнения программы

```
z = lambda w: (w[0] - 1)**2 + (w[1] - 3)**4
res = minimize(z, (0, 0))
res.x
```

```
array([0.99999999, 2.98725136])
```

```
res = minimize(z, (9.999, 3.001))
res.x
```

```
array([1.00000001, 3.001     ])
```

```
z((1, 3)) < z((0.999, 3.001))
```

```
True
```

```
z((1, 3))
```

```
0
```

```
z = lambda w: w[0]**4 + w[1]**4 - 2*w[0]**2 + \
              4*w[0]*w[1] - 2*w[1]**2
z = 4.5*x**(0.33) * y**(0.66)
z_x = diff(z, x)
z_y = diff(z, y)

E_x = (x/z) * z_x
E_y = (y/z) * z_y
print('E_x: %.2f E_y: %.2f' % (E_x, E_y))
```

```
E_x: 0.33 E_y: 0.66
```

```
K, V0 = symbols('K, V0')
V = V0*log(5+K**2)
Vprim2 = diff(V, K, 2)
Vprim3 = diff(V, K, 3)
s = solve(Vprim2, K)
s
```

```
[-sqrt(5), sqrt(5)]
```

```
Vprim3.subs(K, s[1])
```

$$-\frac{\sqrt{5}V_0}{25}$$

Рисунок 11 – Пример выполнения программы

Индивидуальное задание

Индивидуальное задание

```
Найти вторую производную функции y = xe(x2)
```

```python
from sympy import *
x = symbols('x')
y = x*exp(x**2)
diff(y, x, 2)
```

$$2x\left(2x^2+3\right)e^{x^2}$$

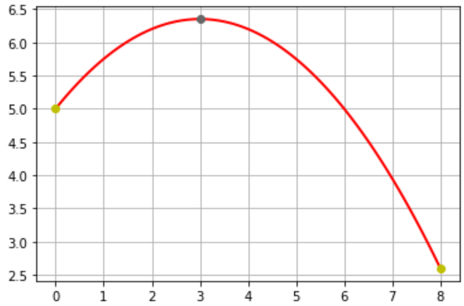
Опытным путем установлено, что объем q продукции, производимой на некотором предприятии одним среднестатистическим рабочим в течение рабочей смены (8 часов), определяется формулой: q = -0.05t^3 + 0.45*t^2 + 5t (0 <= t <=8)

Здесь t – время в часах. Требуется установить зависимость y = y(t) производительности труда рабочего от времени и построить график этой зависимости.

```python
from sympy import *
import numpy as np
import matplotlib.pyplot as plt
t = symbols('t')
q = -0.05*t**3 + 0.45*t**2 + 5*t
y = diff(q, t)
print(y)
```

```
-0.15*t**2 + 0.9*t + 5
```

```python
t = np.linspace(0, 8, 100)
z = -0.15*t**2 + 0.9*t + 5
plt.plot(t, z, '-', lw = 2, color = 'r')
plt.grid(True)
plt.plot([0], [5], 'o', color = 'y')
plt.plot([3], [6.35], 'o', color = '0.4')
plt.plot([8], [2.6], 'o', color = 'y')
plt.show()
```



Графиком этой зависимости y от t является парабола с ветвями, направленными вниз (рис.1). Приведенный график показывает, что производительность труда в первые три часа работы возрастает (с 5 до 6,35 единиц продукции в час), а потом монотонно падает, и к концу рабочего дня составляет лишь 2,6 единиц продукции в час.

Рисунок 12 – Выполнение индивидуального задания