Sprawozdanie Lab10 Marcin Baranowski 78254

Zadanie 1.

Metoda Prostokątów:

```
[3] def integrate(function, a, b, i):
        dx = (b - a) / i
        integr = 0
        for x in range(i):
            x = x * dx + a
            integr += dx * eval(function)
        return integr


    function = input("Funkcja: ")
    a = float(input("Początek przedziału: "))
    b = float(input("Koniec przedziału: "))
    i = int(input("Liczba podprzedziałów: "))
    print("Całka z funkcji {funkcjon} po przedziale od {a} do {b} = {integrate}".format(funkcjon = function, a = a, b = b, integrate = integrate(function, a, b, i)))
```

```
Funkcja: x*x+3
Początek przedziału: 2
Koniec przedziału: 1
Liczba podprzedziałów: 5
Całka z funkcji x*x+3 po przedziale od 2.0 do 1.0 = -5.640000000000001
```

Metoda Trapezów

```
[4] def integrate(function, a, b, i):
        dx = (b - a) / i
        integr = 0
        for x in range(i):
            x = x * dx + a
            fx1 = eval(function)
            x += dx
            fx2 = eval(function)
            integr += 0.5 * dx * (fx1 + fx2)
        return integr

    function = input("Funkcja: ")
    a = float(input("Początek przedziału: "))
    b = float(input("Koniec przedziału: "))
    i = int(input("Liczba podprzedziałów: "))
    print("Całka z funkcji {funkcjon} po przedziale od {a} do {b} = {integrate}".format(funkcjon = function, a = a, b = b, integrate = integrate(function, a, b, i)))
```

```
Funkcja: x*x+2
Początek przedziału: 3
Koniec przedziału: 1
Liczba podprzedziałów: 4
Całka z funkcji x*x+2 po przedziale od 3.0 do 1.0 = -12.75
```

Metoda Simpsona:

```
[5] def integrate(function, a, b, n):
      delta_x = (b-a)/n
      integr = 0
      for i in range(0, n, 2):
        x = a + delta_x * 2 * i
        integr += delta_x * (function(x) + 4 * function(x + delta_x) + function(x + 2 * delta_x)) / 3
      return integr

    a = float(input("Początek przedziału: "))
    b = float(input("Koniec przedziału: "))
    n = int(input("Liczba podprzedziałów: "))
    print("Całka z funkcji x*x po przedziale od {a} do {b} = {integrate}".format(a = a, b = b, integrate = integrate(lambda x: x*x, a, b, n)))
```

```
Początek przedziału: 4
Koniec przedziału: 7
Liczba podprzedziałów: 3
Całka z funkcji x*x po przedziale od 4.0 do 7.0 = 213.33333333333331
```

Zadanie 2.

```
import tensorflow as tf
import numpy as np
from tensorflow import keras

model = tf.keras.Sequential([keras.layers.Dense(units=1, input_shape=[1])])

model.compile(optimizer='sgd', loss='mean_squared_error')

xs = np.array([ 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0], dtype=float)
ys = np.array([ 6.0, 0.0, -4.0, 0.0, 18.0, 56.0, 120.0, 216.0], dtype=float)

model.fit(xs, ys, epochs=5000)
```

Strumieniowane dane wyjściowe obcięte do 5000 ostatnich wierszy.

```
Epoch 2501/5000
1/1 [==============================] - 0s 6ms/step - loss: 1591.5000
Epoch 2502/5000
1/1 [==============================] - 0s 8ms/step - loss: 1591.5001
Epoch 2503/5000
1/1 [==============================] - 0s 6ms/step - loss: 1591.4999
Epoch 2504/5000
1/1 [==============================] - 0s 7ms/step - loss: 1591.4999
Epoch 2505/5000
1/1 [==============================] - 0s 5ms/step - loss: 1591.5001
Epoch 2506/5000
1/1 [==============================] - 0s 6ms/step - loss: 1591.5001
Epoch 2507/5000
1/1 [==============================] - 0s 7ms/step - loss: 1591.4998
Epoch 2508/5000
1/1 [==============================] - 0s 8ms/step - loss: 1591.5000
Epoch 2509/5000
1/1 [==============================] - 0s 7ms/step - loss: 1591.5000
Epoch 2510/5000
1/1 [==============================] - 0s 8ms/step - loss: 1591.5000
Epoch 2511/5000
1/1 [==============================] - 0s 7ms/step - loss: 1591.4999
Epoch 2512/5000
1/1 [==============================] - 0s 11ms/step - loss: 1591.4998
Epoch 2513/5000
1/1 [==============================] - 0s 10ms/step - loss: 1591.4999
Epoch 2514/5000
1/1 [==============================] - 0s 4ms/step - loss: 1591.5001
Epoch 2515/5000
1/1 [==============================] - 0s 5ms/step - loss: 1591.5001
```

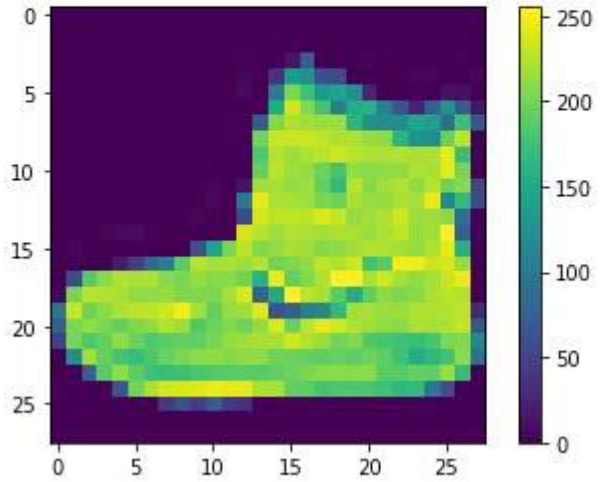
Zadanie 3.

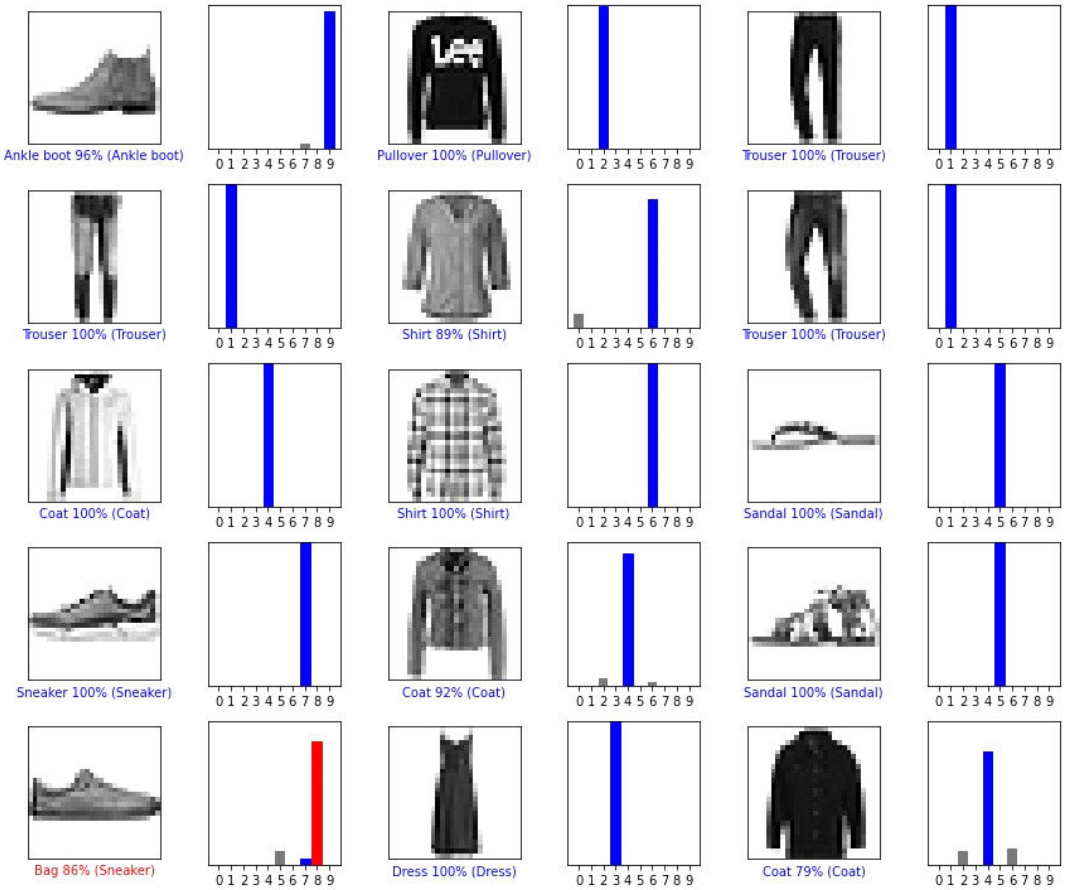
```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt

fashion_mnist = tf.keras.datasets.fashion_mnist

(train_images, train_labels), (test_images, test_labels) = fashion_mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-labels-idx1-ubyte.gz
32768/29515 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-images-idx3-ubyte.gz
26427392/26421880 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-labels-idx1-ubyte.gz
8192/5148 [===============================================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-images-idx3-ubyte.gz
4423680/4422102 [==============================] - 0s 0us/step
```

```
class_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat',
               'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot']
```

```
plt.figure()
plt.imshow(train_images[0])
plt.colorbar()
plt.grid(False)
plt.show()
```




Ankle boot 96% (Ankle boot)
Pullover 100% (Pullover)
Trouser 100% (Trouser)
Trouser 100% (Trouser)
Shirt 89% (Shirt)
Trouser 100% (Trouser)
Coat 100% (Coat)
Shirt 100% (Shirt)
Sandal 100% (Sandal)
Sneaker 100% (Sneaker)
Coat 92% (Coat)
Sandal 100% (Sandal)
Bag 86% (Sneaker)
Dress 100% (Dress)
Coat 79% (Coat)
0 1 2 3 4 5 6 7 8 9