TASK 1-2

```
dna_seq = input("Podaj sekwencję DNA: ")

# Zainicjowanie liczników nukleotydów
licznik_adeniny = 0
licznik_cytozyny = 0
licznik_guaniny = 0
licznik_tyminy = 0

# Przejście przez każdy nukleotyd w sekwencji DNA
for nukleotyd in dna_seq:
    if nukleotyd == "A":
        licznik_adeniny += 1
    elif nukleotyd == "C":
        licznik_cytozyny += 1
    elif nukleotyd == "G":
        licznik_guaniny += 1
    elif nukleotyd == "T":
        licznik_tyminy += 1

print("Liczba adenin: ", licznik_adeniny)
print("Liczba cytozyn: ", licznik_cytozyny)
print("Liczba guanin: ", licznik_guaniny)
print("Liczba tymin: ", licznik_tyminy)
```

TASK 3-5

```
sekwencja_dna = input("Podaj sekwencję DNA: ")

# Podmiana wystąpień tyminy i uracylu
sekwencja_rna = sekwencja_dna.replace("T", "U")

print("Sekwencja RNA: ", sekwencja_rna)
print("Długość sekwencji RNA: ", len(sekwencja_rna))
```

TASK 6-8

```python
# Słownik zawierający masy aminokwasów
amino_acid_masses = {
    "A": 89.094,
    "C": 121.154,
    "D": 133.104,
    "E": 147.131,
    "F": 165.192,
    "G": 75.067,
    "H": 155.156,
    "I": 131.175,
    "K": 146.189,
    "L": 131.175,
    "M": 149.208,
    "N": 132.119,
    "P": 115.132,
    "Q": 146.146,
    "R": 174.203,
    "S": 105.093,
    "T": 119.120,
    "V": 117.148,
    "W": 204.228,
    "Y": 181.191,
}

peptyd = input("Podaj sekwencję peptydu: ")

# Masa cząsteczkowa peptydu
masa = sum([amino_acid_masses[a] for a in peptyd])

# Wynik
print("Masa cząsteczkowa peptydu: ", round(masa, 2))
```

TASK 9-10

```python
sekwencja_dna = "ATGCTACGATCGTACGGCGCAAATAGCTAGCTAGCTAGC"

# Liczba par zasad GC w sekwencji
liczba_gc = sekwencja_dna.count("G") + sekwencja_dna.count("C")

# Procentowej zawartość par zasad GC
procent_gc = liczba_gc / len(sekwencja_dna) * 100

# Wyświetlenie wyniku
print("Procentowa zawartość par zasad GC: {:.2f}%".format(procent_gc))
```

TASK 11-12

```
# Dane wejściowe
homozygoty_dominujace = 100
heterozygoty = 200
homozygoty_recesywne = 100
# dalej już Twój kod

# Liczba alleli
liczba_alleli = homozygoty_dominujace + heterozygoty + homozygoty_recesywne

# Występowanie allelu dominującego (A)
czestotliwosc_a = (2 * homozygoty_dominujace +
                   heterozygoty) / (2 * liczba_alleli)

# Występowanie allelu recesywnego (a)
czestotliwosc_A = 1 - czestotliwosc_a

# Wyświetlenie wyniku
print("Częstotliwość dla allelu A:", "{:.2f}".format(czestotliwosc_a))
print("Częstotliwość dla allelu a:", "{:.2f}".format(czestotliwosc_A))
```

TASK 13-15

Dane wejściowe:

Sekwencja	Własność1	Własność2
AAKLPLAR	1,2	3,5
KKLPARAA	0,8	3,1
LLPKARAA	1,1	2,9
LARPKKAA	1,4	4,2
KKALPRRA	0,9	3,8
RLPKALAA	1,6	4
LARAKKPA	1,3	3,3
KAAARPLR	1	2,7
ALRARKPA	1,5	3,9
RALKLPKA	1,7	4,1

```python
import pandas as pd

break_line = "\n" + 50*"-" + "\n"

# Wczytanie danych z pliku Excel
dataframe = pd.read_excel('Krzysztof_Lipski_lab_9_i_10/dane_sekwencji.xlsx')

# Obliczenie statystyk dla każdej właściwości
srednia_w1 = dataframe['Własność1'].mean()
mediana_w1 = dataframe['Własność1'].median()
odchylenie_std_w1 = dataframe['Własność1'].std()

# Obliczenie statystyk dla każdej właściwości
srednia_w2 = dataframe['Własność2'].mean()
mediana_w2 = dataframe['Własność2'].median()
odchylenie_std_w2 = dataframe['Własność2'].std()

# Wyświetlenie wyników
print("Średnia własność 1:" , srednia_w1)
print("Mediana własność 1:", mediana_w1)
print("Odchylenie standardowe własność 1:", odchylenie_std_w1)

print(break_line)

print("Średnia własność 2:" , srednia_w2)
print("Mediana własność 2:", mediana_w2)
print("Odchylenie standardowe własność 2:", odchylenie_std_w2)
```

TASK 16-18

Dane wejściowe

```
Grupa wiekowa,Liczba zaszczepionych
0-17,150000
18-29,500000
30-49,750000
50-64,850000
65+,900000ty
```

Rozwiązanie

```
import pandas as pd
import matplotlib.pyplot as plt

# Wczytanie danych z pliku CSV
dane = pd.read_csv('Krzysztof_Lipski_lab_9_i_10/dane_szczepienia.csv')

# Stworzenie wykresu słupkowego
plt.bar(dane['Grupa wiekowa'], dane['Liczba zaszczepionych'])

# Dodanie tytułu wykresu i opisu osi x i y
plt.title('Liczba osób, które otrzymay szczepienie przeciwko COVID-19 w różnych
grupach wiekowych')
plt.xlabel('Grupa wiekowa')
plt.ylabel('Liczba zaszczepionych')

plt.show()
```

Wynik

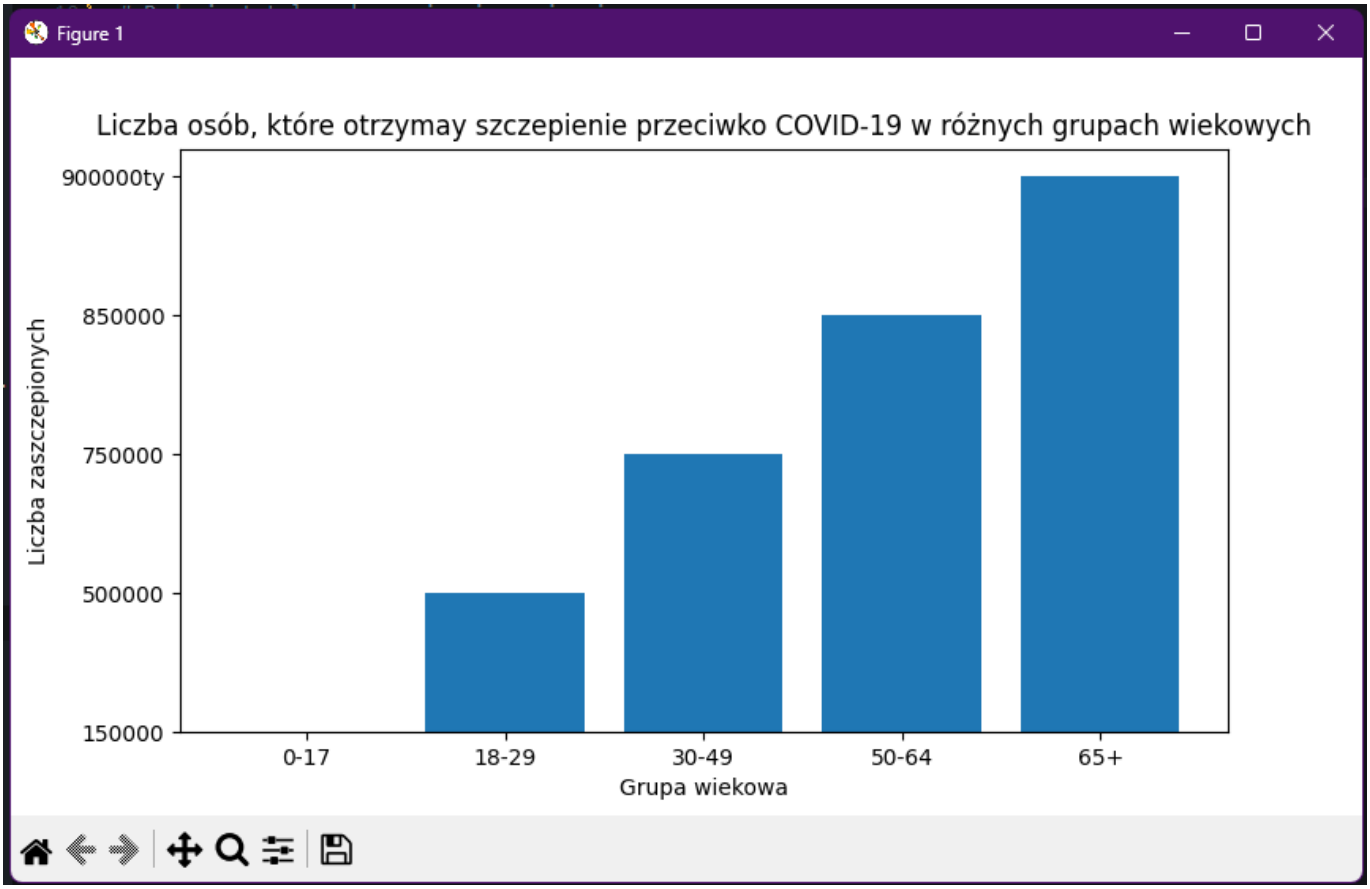
Figure 1
Liczba osób, które otrzymay szczepienie przeciwko COVID-19 w różnych grupach wiekowych
900000ty
850000
750000
500000
150000
Liczba zaszczepionych
0-17
18-29
30-49
50-64
65+
Grupa wiekowa

TASK 19-21

```python
import pandas as pd
from scipy.stats import ttest_ind

# Wczytanie danych
dane1 = pd.read_csv('Krzysztof_Lipski_lab_9_i_10/dane_biologiczne1.csv')
dane2 = pd.read_csv('Krzysztof_Lipski_lab_9_i_10/dane_biologiczne2.csv')

# Obliczenie średniej i odchylenia standardowego dla każdego zbioru danych
srednia1 = dane1['wartosc'].mean()
odchylenie1 = dane1['wartosc'].std()

srednia2 = dane2['wartosc'].mean()
odchylenie2 = dane2['wartosc'].std()

# Przeprowadzenie t-test dla dwóch niezależnych prób
statystyka_t, p_value = ttest_ind(dane1['wartosc'], dane2['wartosc'])

print('Średnia wartość dla pierwszego zbioru danych:', srednia1)
print('Odchylenie standardowe dla pierwszego zbioru danych:', odchylenie1)

print('Średnia wartość dla drugiego zbioru danych:', srednia2)
print('Odchylenie standardowe dla drugiego zbioru danych:', odchylenie2)

print('Statystyka t:', statystyka_t)
print('P-wartość:', p_value)
```

TASK 22-24

```python
import Bio
from Bio.Seq import Seq

# Wczytanie sekwencję DNA za pomocą funkcji input()
seq_dna = input('Wprowadź sekwencję DNA: ')

# Transkrypcja sekwencji DNA na sekwencję RNA
seq_rna = Seq(seq_dna).transcribe()

# Translacja sekwencji RNA na łańcuch białkowy
seq_protein = seq_rna.translate()

print('Sekwencja DNA:', seq_dna)
print('Sekwencja RNA:', seq_rna)
print('Łańcuch białkowy:', seq_protein)
```

TASK 25-29

```python
from Bio import Entrez

Entrez.email = "s20901@pjwstk.edu.pl"

handle = Entrez.esearch(db="pubmed", term="colon cancer", retmax=10)
record = Entrez.read(handle)

handle = Entrez.efetch(db="pubmed", id=record["IdList"],
                       rettype="medline", retmode="text")
records = handle.read()

for record in records.split("\n\n"):
    if record.strip():
        title = ""
        authors = ""
        date = ""

        for line in record.split("\n"):
            if line.startswith("TI"):
                title = line[6:].strip()
            elif line.startswith("AU"):
                authors = line[6:].strip()
            elif line.startswith("DP"):
                date = line[6:].strip()

        print("Tytuł:", title)
        print("Autorzy:", authors)
        print("Data publikacji:", date, '\n')
```