

# Laboratorium 1

Autorzy: Krzysztof Zalewa 273032, Michał Pakuła 272828 Data: 24 Marca 2025

## Ćwiczenie 1

Celem ćwiczenia było:

1. Napisanie skryptu w Pythonie umożliwiającego wczytywanie i wizualizację badanych sygnałów.
  - ekg1.txt – 12 kolumn odpowiadających odprowadzeniom,  $f_s = 1000$  Hz
  - ekg100.txt – 1 kolumna,  $f_s = 360$  Hz
  - ekg\_noise.txt – 1 kolumna: czas, 2 kolumna: wartości amplitud EKG,  $f_s = 360$  Hz
2. Umożliwienie obserwacji wycinka sygnału

```
In [1]: import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

```
In [2]: # Ładowanie wybranego pliku
file_name = ".../src//"+input("Podaj nazwę pliku z danymi: ")

data_frame = pd.read_csv(file_name, sep="\s+", header=None, engine="python")
```

Po załadowaniu pliku .txt są trzy możliwości

1. Plik ma 12 kolumn z danymi
2. Plik ma 2 kolumny z danymi
3. Plik ma 1 kolumnę z danymi

```
In [3]: # Sprawdź liczbę kolumn
num_rows = len(data_frame)
start = input("Początek zakresu(Minimalnie 0): ")
end = input("Koniec zakresu(Maksymalnie "+str(num_rows)+"): ")
tick_rate = 72000
if start == "":
    start = 0
if end == "":
    end = num_rows

x = list(range(int(start),int(end)))
new_data = data_frame.iloc[int(start):int(end)].copy()
font = {'size':20}
num_rows = len(new_data)
minutes = num_rows/tick_rate
x_labels = np.linspace(0,minutes,int(minutes))

if len(data_frame.columns) == 12:
    column_names = ['I','II','III','aVL','aVR','aVF','V_1','V_2','V_3','V_4','V_5','V_6']
    new_data.columns = column_names
    plt.figure(figsize=(50,15))
    j=0
    for i in range(1,12,2):
        plt.subplot(6,2,i)
        plt.plot(x,new_data[column_names[j]])
        plt.grid(True)
        plt.xticks(ticks=x_labels*tick_rate,labels=x_labels)
        plt.title(column_names[j],fontdict=font)
        j += 1
    for i in range(2,13,2):
        plt.subplot(6,2,i)
        plt.plot(x,new_data[column_names[j]])
        plt.xticks(ticks=x_labels*tick_rate,labels=x_labels)
        plt.grid(True)
        plt.title(column_names[j],fontdict=font)
        j += 1
    plt.tight_layout()
    plt.show()

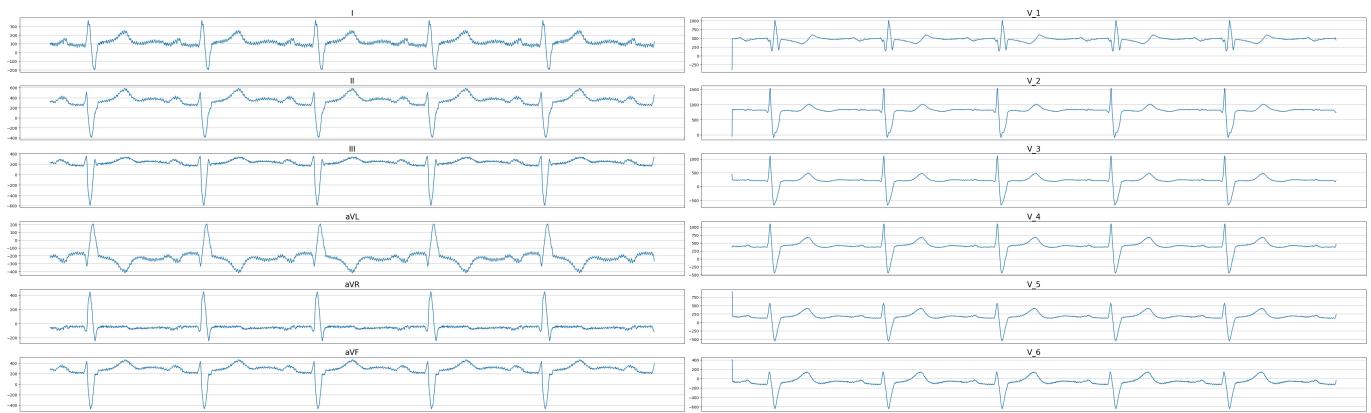
if len(new_data.columns) == 1:
    new_data.columns = ['data']
    font = {'size':20}
    plt.figure(figsize=(20,5))
    plt.plot(x,new_data['data'])
    plt.xticks(ticks=x_labels*tick_rate,labels=x_labels)
```

```

plt.grid(True)
plt.title("EKG")
plt.show()

if len(new_data.columns) == 2:
    new_data.columns = ['I','data']
    font = {'size':20}
    plt.figure(figsize=(20,5))
    plt.plot(x,new_data['data'])
    plt.xticks(ticks=x_labels*tick_rate,labels=x_labels)
    plt.grid(True)
    plt.title("EKG")

```



Powyżej znajduję się wykres załadowany EKG dla pliku ekg1.txt (Dla plików ekg100.txt i ekg\_noise.txt trzeba zmniejszyć zakres danych by wykres był bardziej czytelny)

## Ćwiczenie 2

Celem ćwiczenia było:

1. Wygeneruj ciąg próbek odpowiadający falę sinusoidalnej o częstotliwości 50 Hz i długości 65536.
2. Wyznacz dyskretną transformatę Fouriera tego sygnału i przedstaw jego widmo amplitudowe na wykresie w zakresie częstotliwości [0, fs/2], gdzie fs oznacza częstotliwość próbkowania.
3. Wygeneruj ciąg próbek mieszaniny dwóch fal sinusoidalnych (tzn. ich kombinacji liniowej) o częstotliwościach 50 i 60 Hz. Wykonaj zadanie z punktu 2 dla tego sygnału.
4. Powtórz eksperymenty dla różnych czasów trwania sygnałów, tzn. dla różnych częstotliwości próbkowania.
5. Wyznacz odwrotne transformaty Fouriera ciągów wyznaczonych w zadaniu 2 i porównaj z ciągami oryginalnymi.

```
In [50]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
In [51]: # Generacja próbek

fs = 44100
freq1 = 50
freq2 = 60
length = 65536

t = np.arange(length) / fs
sin_wave1 = np.sin(2 * np.pi * freq1 * t)

t = np.arange(length) / fs
sin_wave2 = np.sin(2 * np.pi * freq2 * t)

mixed_wave = sin_wave1 + sin_wave2
```

### Zadanie nr1

Jednocześnie generowana jest fala sinusoidalna do zadania 2 oraz mieszanina próbek do zadania 3.

```
In [ ]: # Wyświetlenie i transformata F

plt.figure(figsize=(20,5))
plt.subplot(4,1,1)
plt.plot(sin_wave1)
plt.grid(True)
plt.title("Sygnał przed transformata")

fourier1 = np.fft.fft(sin_wave1)

widmo = np.abs(fourier1)
abs_widmo = widmo / np.max(widmo)
freq = np.fft.fftfreq(len(t),1/fs)
pos_freq = freq[:len(freq)//2]
pos_widmo = abs_widmo[:len(abs_widmo)//2]

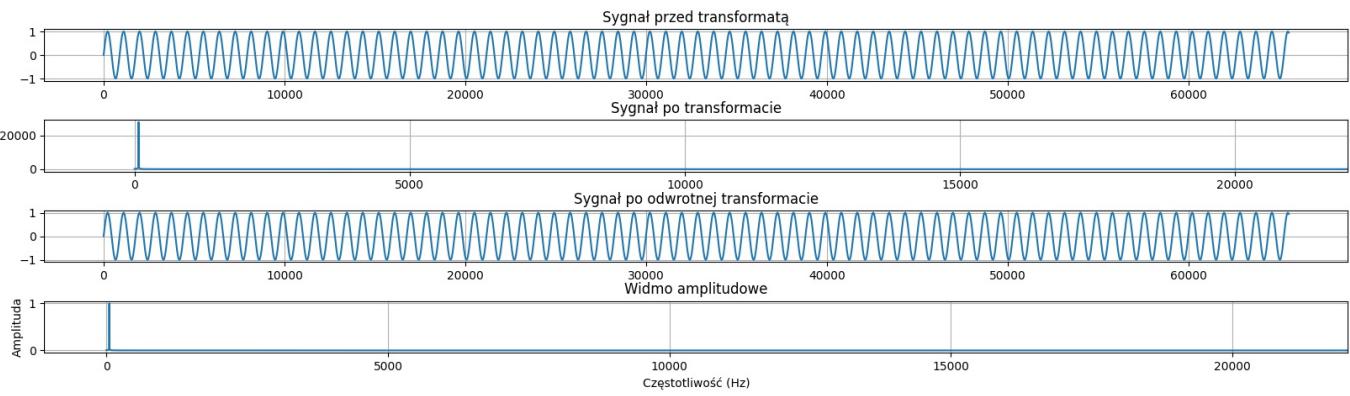
plt.subplot(4,1,2)
plt.plot(widmo[:length//2])
plt.xlim(right = fs/2)
plt.grid(True)
plt.title("Sygnał po transformacie")

fourier2 = np.fft.ifft(fourier1)

plt.subplot(4,1,3)
plt.plot(fourier2)
plt.grid(True)
plt.title("Sygnał po odwrotnej transformacie")

plt.subplot(4,1,4)
plt.plot(pos_freq,pos_widmo)
plt.xlim(right = fs/2)
plt.grid(True)
plt.title("Widmo amplitudowe")
plt.xlabel('Częstotliwość (Hz)')
plt.ylabel('Amplituda')

plt.subplots_adjust(hspace=0.75)
plt.show()
```



## Zadanie nr2

Przy użyciu biblioteki np.fft tworzona jest transformata fouriera i widmo sygnału.

```
In [ ]: #
plt.figure(figsize=(20,5))
plt.subplot(4,1,1)
plt.plot(mixed_wave)
plt.grid(True)
plt.title("Sygnał przed transformata")

fourier1 = np.fft.fft(mixed_wave)
widmo = np.abs(fourier1)
abs_widmo = widmo / np.max(widmo)
freq = np.fft.fftfreq(len(t),1/fs)
pos_freq = freq[:len(freq)//2]
pos_widmo = abs_widmo[:len(abs_widmo)//2]

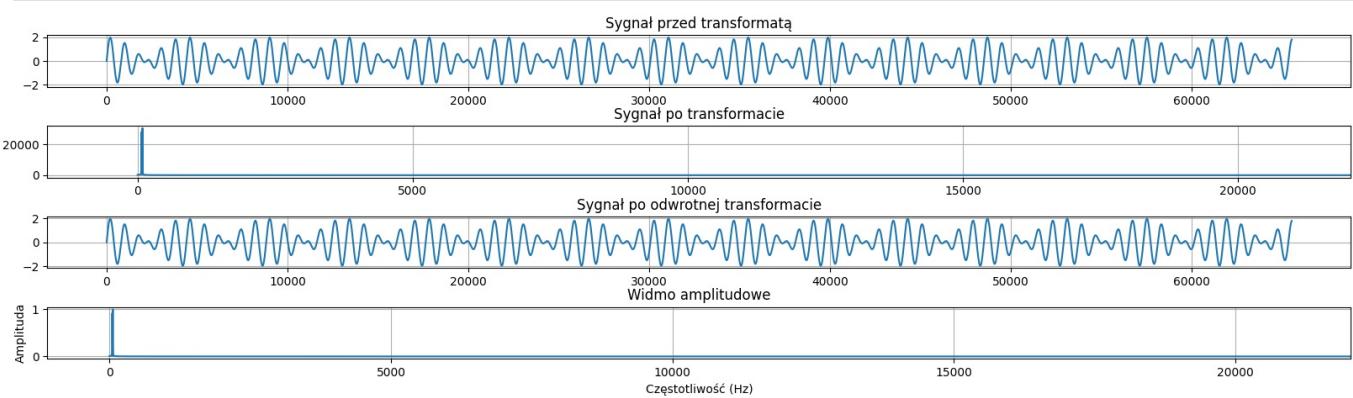
plt.subplot(4,1,2)
plt.plot(widmo[:length//2])
plt.xlim(right = fs/2)
plt.grid(True)
plt.title("Sygnał po transformacie")

fourier2 = np.fft.ifft(fourier1)

plt.subplot(4,1,3)
plt.plot(fourier2)
plt.grid(True)
plt.title("Sygnał po odwrotnej transformacie")

plt.subplot(4,1,4)
plt.plot(pos_freq,pos_widmo)
plt.xlim(right = fs/2)
plt.grid(True)
plt.title("Widmo amplitudowe")
plt.xlabel('Częstotliwość (Hz)')
plt.ylabel('Amplituda')

plt.subplots_adjust(hspace=0.75)
plt.show()
```



## Zadanie nr3

Zadanie nr3 jest powtórką zadania nr 2 więc kod będzie w zasadzie taki sam (różnicą jest sygnał wykorzystany do w transformacie fouriera).

## Zadanie nr5

Zadanie nr5 zostało rozwiążane odpowiednio w zadaniu nr2 i nr3.

## Ćwiczenie 1

Celem ćwiczenia było:

1. Napisanie skryptu w Pythonie umożliwiającego wczytywanie i wizualizację badanych sygnałów.
  - ekg1.txt – 12 kolumn odpowiadających odprowadzeniom, fs = 1000 Hz
  - ekg100.txt – 1 kolumna, fs = 360 Hz
  - ekg\_noise.txt – 1 kolumna: czas, 2 kolumna: wartości amplitud EKG, fs = 360 Hz
2. Umożliwienie obserwacji wycinka sygnału

```
In [1]: import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

```
In [2]: # Ładowanie wybranego pliku
file_name = ".../src//"+input("Podaj nazwę pliku z danymi: ")

data_frame = pd.read_csv(file_name, sep="\s+", header=None, engine="python")
```

Po załadowaniu pliku .txt są trzy możliwości

1. Plik ma 12 kolumn z danymi
2. Plik ma 2 kolumny z danymi
3. Plik ma 1 kolumnę z danymi

```
In [3]: # Sprawdź liczbę kolumn
num_rows = len(data_frame)
start = input("Początek zakresu(Minimalnie 0): ")
end = input("Koniec zakresu(Maksymalnie "+str(num_rows)+": ")
tick_rate = 72000
if start == "":
    start = 0
if end == "":
    end = num_rows

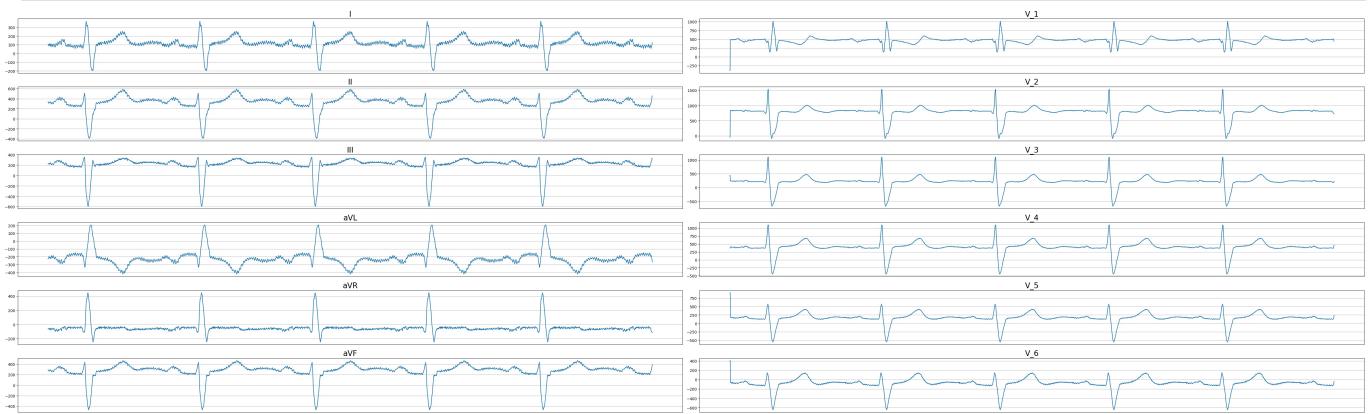
x = list(range(int(start),int(end)))
new_data = data_frame.iloc[int(start):int(end)].copy()
font = {'size':20}
num_rows = len(new_data)
minutes = num_rows/tick_rate
x_labels = np.linspace(0,minutes,int(minutes))

if len(data_frame.columns) == 12:
    column_names = ['I', 'II', 'III', 'aVL', 'aVR', 'aVF', 'V_1', 'V_2', 'V_3', 'V_4', 'V_5', 'V_6']
    new_data.columns = column_names
    plt.figure(figsize=(50,15))
    j=0
    for i in range(1,12,2):
        plt.subplot(6,2,i)
        plt.plot(x,new_data[column_names[j]])
        plt.grid(True)
        plt.xticks(ticks=x_labels*tick_rate,labels=x_labels)
        plt.title(column_names[j],fontdict=font)
        j += 1
    for i in range(2,13,2):
        plt.subplot(6,2,i)
        plt.plot(x,new_data[column_names[j]])
        plt.xticks(ticks=x_labels*tick_rate,labels=x_labels)
        plt.grid(True)
        plt.title(column_names[j],fontdict=font)
        j += 1
    plt.tight_layout()
    plt.show()

if len(new_data.columns) == 1:
    new_data.columns = ['data']
    font = {'size':20}
    plt.figure(figsize=(20,5))
    plt.plot(x,new_data['data'])
    plt.xticks(ticks=x_labels*tick_rate,labels=x_labels)
    plt.grid(True)
    plt.title("EKG")
    plt.show()

if len(new_data.columns) == 2:
    new_data.columns = ['I','data']
```

```
font = {'size':20}
plt.figure(figsize=(20,5))
plt.plot(x,new_data['data'])
plt.xticks(ticks=x_labels*tick_rate,labels=x_labels)
plt.grid(True)
plt.title("EKG")
```



Powyżej znajduję się wykres załadowany EKG dla pliku ekg1.txt (Dla plików ekg100.txt i ekg\_noise.txt trzeba zmniejszyć zakres danych by wykres był bardziej czytelny)

## Ćwiczenie 2

Celem ćwiczenia było:

1. Wygeneruj ciąg próbek odpowiadający falę sinusoidalnej o częstotliwości 50 Hz i długości 65536.
2. Wyznacz dyskretną transformatę Fouriera tego sygnału i przedstaw jego widmo amplitudowe na wykresie w zakresie częstotliwości [0, fs/2], gdzie fs oznacza częstotliwość próbkowania.
3. Wygeneruj ciąg próbek mieszaniny dwóch fal sinusoidalnych (tzn. ich kombinacji liniowej) o częstotliwościach 50 i 60 Hz. Wykonaj zadanie z punktu 2 dla tego sygnału.
4. Powtórz eksperymenty dla różnych czasów trwania sygnałów, tzn. dla różnych częstotliwości próbkowania.
5. Wyznacz odwrotne transformaty Fouriera ciągów wyznaczonych w zadaniu 2 i porównaj z ciągami oryginalnymi.

```
In [50]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
In [51]: # Generacja próbek

fs = 44100
freq1 = 50
freq2 = 60
length = 65536

t = np.arange(length) / fs
sin_wave1 = np.sin(2 * np.pi * freq1 * t)

t = np.arange(length) / fs
sin_wave2 = np.sin(2 * np.pi * freq2 * t)

mixed_wave = sin_wave1 + sin_wave2
```

### Zadanie nr1

Jednocześnie generowana jest fala sinusoidalna do zadania 2 oraz mieszanina próbek do zadania 3.

```
In [ ]: # Wyświetlenie i transformata F

plt.figure(figsize=(20,5))
plt.subplot(4,1,1)
plt.plot(sin_wave1)
plt.grid(True)
plt.title("Sygnał przed transformata")

fourier1 = np.fft.fft(sin_wave1)

widmo = np.abs(fourier1)
abs_widmo = widmo / np.max(widmo)
freq = np.fft.fftfreq(len(t),1/fs)
pos_freq = freq[:len(freq)//2]
pos_widmo = abs_widmo[:len(abs_widmo)//2]

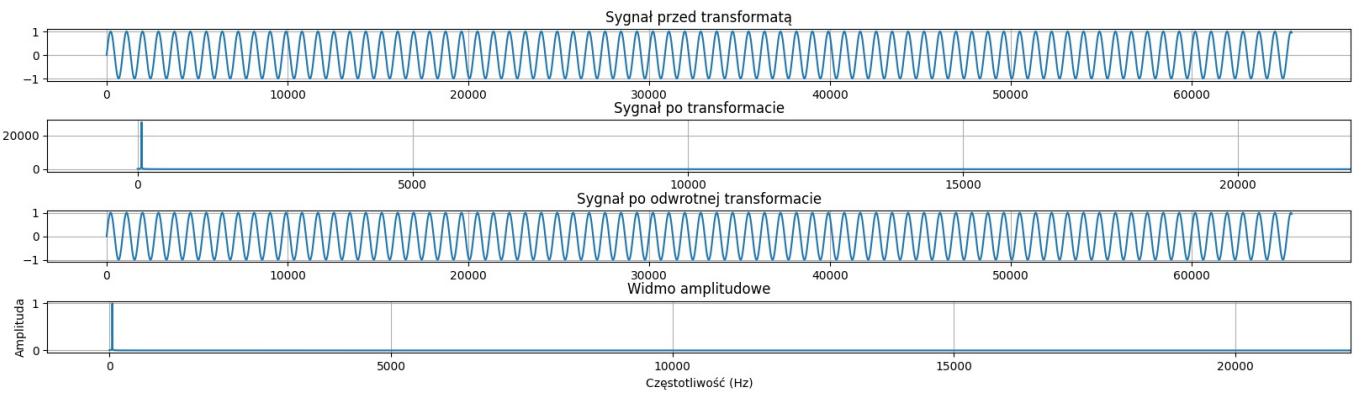
plt.subplot(4,1,2)
plt.plot(widmo[:length//2])
plt.xlim(right = fs/2)
plt.grid(True)
plt.title("Sygnał po transformacie")

fourier2 = np.fft.ifft(fourier1)

plt.subplot(4,1,3)
plt.plot(fourier2)
plt.grid(True)
plt.title("Sygnał po odwrotnej transformacie")

plt.subplot(4,1,4)
plt.plot(pos_freq,pos_widmo)
plt.xlim(right = fs/2)
plt.grid(True)
plt.title("Widmo amplitudowe")
plt.xlabel('Częstotliwość (Hz)')
plt.ylabel('Amplituda')

plt.subplots_adjust(hspace=0.75)
plt.show()
```



## Zadanie nr2

Przy użyciu biblioteki np.fft tworzona jest transformata fouriera i widmo sygnału.

```
In [ ]: #
plt.figure(figsize=(20,5))
plt.subplot(4,1,1)
plt.plot(mixed_wave)
plt.grid(True)
plt.title("Sygnał przed transformata")

fourier1 = np.fft.fft(mixed_wave)
widmo = np.abs(fourier1)
abs_widmo = widmo / np.max(widmo)
freq = np.fft.fftfreq(len(t),1/fs)
pos_freq = freq[:len(freq)//2]
pos_widmo = abs_widmo[:len(abs_widmo)//2]

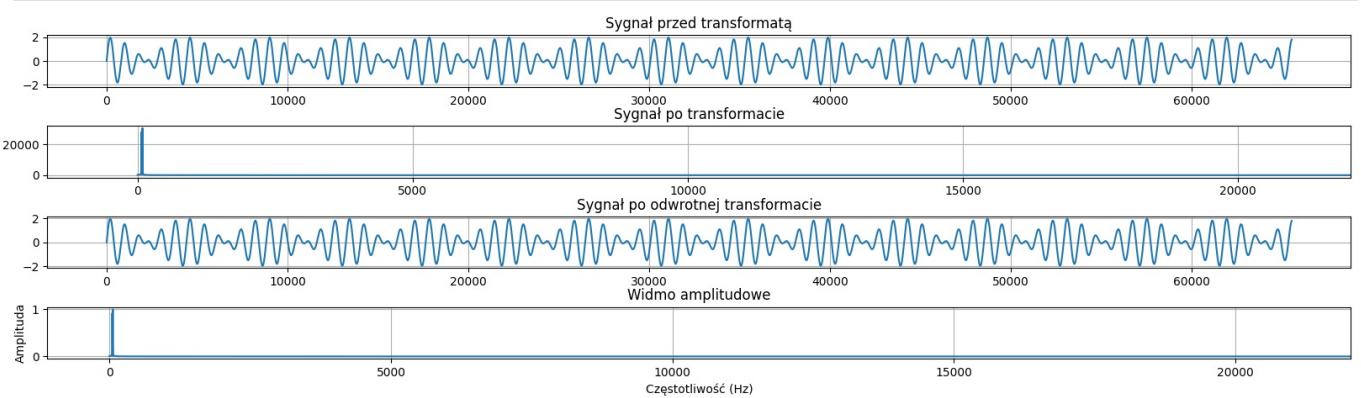
plt.subplot(4,1,2)
plt.plot(widmo[:length//2])
plt.xlim(right = fs/2)
plt.grid(True)
plt.title("Sygnał po transformacie")

fourier2 = np.fft.ifft(fourier1)

plt.subplot(4,1,3)
plt.plot(fourier2)
plt.grid(True)
plt.title("Sygnał po odwrotnej transformacie")

plt.subplot(4,1,4)
plt.plot(pos_freq,pos_widmo)
plt.xlim(right = fs/2)
plt.grid(True)
plt.title("Widmo amplitudowe")
plt.xlabel('Częstotliwość (Hz)')
plt.ylabel('Amplituda')

plt.subplots_adjust(hspace=0.75)
plt.show()
```



## Zadanie nr3

Zadanie nr3 jest powtórką zadania nr 2 więc kod będzie w zasadzie taki sam (różnicą jest sygnał wykorzystany do w transformacie fouriera).

## Zadanie nr5

Zadanie nr5 zostało rozwiążane odpowiednio w zadaniu nr2 i nr3.

# Ćwiczenie 1

Celem ćwiczenia było:

1. Napisanie skryptu w Pythonie umożliwiającego wczytywanie i wizualizację badanych sygnałów.

- ekg1.txt – 12 kolumn odpowiadających odczytanym,  $f_s = 1000$  Hz
- ekg100.txt – 1 kolumna,  $f_s = 360$  Hz
- ekg\_noise.txt – 1 kolumna: czas, 2 kolumna: wartości amplitud EKG,  $f_s = 360$  Hz

2. Umożliwienie obserwacji wycinka sygnału

```
In [1]: import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

```
In [2]: # Ładowanie wybranego pliku
file_name = "../src//"+input("Podaj nazwę pliku z danymi: ")

data_frame = pd.read_csv(file_name, sep="\s+", header=None, engine="python")
```

Po załadowaniu pliku .txt są trzy możliwości

1. Plik ma 12 kolumn z danymi
2. Plik ma 2 kolumny z danymi
3. Plik ma 1 kolumnę z danymi

```
In [3]: # Sprawdź liczbę kolumn
num_rows = len(data_frame)
start = input("Początek zakresu(Minimalnie 0): ")
end = input("Koniec zakresu(Maksymalnie "+str(num_rows)+"): ")
tick_rate = 72000
if start == "":
    start = 0
if end == "":
    end = num_rows

x = list(range(int(start),int(end)))
new_data = data_frame.iloc[int(start):int(end)].copy()
font = {'size':20}
num_rows = len(new_data)
minutes = num_rows/tick_rate
x_labels = np.linspace(0,minutes,int(minutes))

if len(data_frame.columns) == 12:
    column_names = ['I','II','III','aVL','aVR','aVF','V_1','V_2','V_3','V_4','V_5','V_6']
    new_data.columns = column_names
    plt.figure(figsize=(50,15))
    j=0
    for i in range(1,12,2):
        plt.subplot(6,2,i)
        plt.plot(x,new_data[column_names[j]])
        plt.grid(True)
```

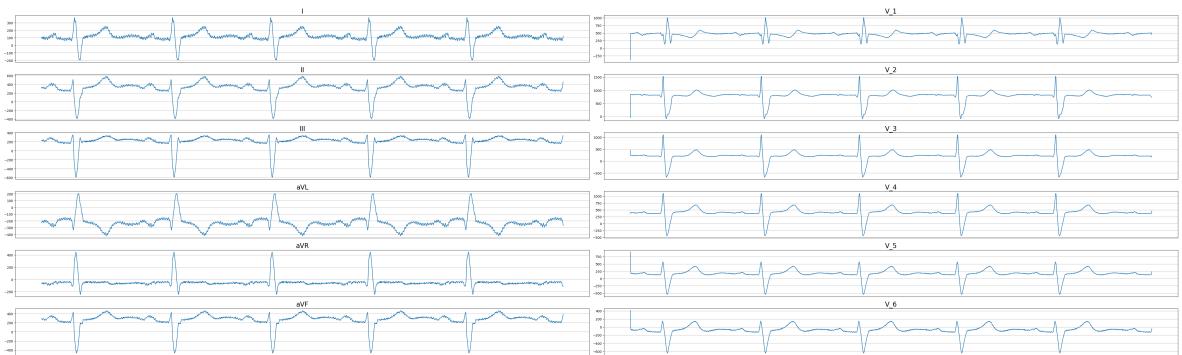
```

plt.xticks(ticks=x_labels*tick_rate,labels=x_labels)
plt.title(column_names[j],fontdict=font)
j += 1
for i in range(2,13,2):
    plt.subplot(6,2,i)
    plt.plot(x,new_data[column_names[j]])
    plt.xticks(ticks=x_labels*tick_rate,labels=x_labels)
    plt.grid(True)
    plt.title(column_names[j],fontdict=font)
    j += 1
plt.tight_layout()
plt.show()

if len(new_data.columns) == 1:
    new_data.columns = ['data']
    font = {'size':20}
    plt.figure(figsize=(20,5))
    plt.plot(x,new_data['data'])
    plt.xticks(ticks=x_labels*tick_rate,labels=x_labels)
    plt.grid(True)
    plt.title("EKG")
    plt.show()

if len(new_data.columns) == 2:
    new_data.columns = ['I','data']
    font = {'size':20}
    plt.figure(figsize=(20,5))
    plt.plot(x,new_data['data'])
    plt.xticks(ticks=x_labels*tick_rate,labels=x_labels)
    plt.grid(True)
    plt.title("EKG")

```



Powyżej znajduję się wykres załadowany EKG dla pliku ekg1.txt (Dla plików ekg100.txt i ekg\_noise.txt trzeba zmniejszyć zakres danych by wykres był bardziej czytelny)

## Ćwiczenie 2

Celem ćwiczenia było:

1. Wygeneruj ciąg próbek odpowiadający fali sinusoidalnej o częstotliwości 50 Hz i długości 65536.
2. Wyznacz dyskretną transformatę Fouriera tego sygnału i przedstaw jego widmo amplitudowe na wykresie w zakresie częstotliwości  $[0, fs/2]$ , gdzie  $fs$  oznacza częstotliwość próbkowania.
3. Wygeneruj ciąg próbek mieszaniny dwóch fal sinusoidalnych (tzn. ich kombinacji liniowej) o częstotliwościach 50 i 60 Hz. Wykonaj zadanie z punktu 2 dla tego sygnału.
4. Powtórz eksperymenty dla różnych czasów trwania sygnałów, tzn. dla różnych częstotliwości próbkowania.
5. Wyznacz odwrotne transformaty Fouriera ciągów wyznaczonych w zadaniu 2 i porównaj z ciągami oryginalnymi.

```
In [50]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
In [51]: # Generacja próbek

fs = 44100
freq1 = 50
freq2 = 60
length = 65536

t = np.arange(length) / fs
sin_wave1 = np.sin(2 * np.pi * freq1 * t)

t = np.arange(length) / fs
sin_wave2 = np.sin(2 * np.pi * freq2 * t)

mixed_wave = sin_wave1 + sin_wave2
```

### Zadanie nr1

Jednocześnie generowana jest fala sinusoidalna do zadania 2 oraz mieszanina próbek do zadania 3.

```
In [ ]: # Wyświetlenie i transformata F

plt.figure(figsize=(20,5))
plt.subplot(4,1,1)
plt.plot(sin_wave1)
plt.grid(True)
plt.title("Sygnał przed transformata")

fourier1 = np.fft.fft(sin_wave1)
```

```

widmo = np.abs(fourier1)
abs_widmo = widmo / np.max(widmo)
freq = np.fft.fftfreq(len(t), 1/fs)
pos_freq = freq[:len(freq)//2]
pos_widmo = abs_widmo[:len(abs_widmo)//2]

plt.subplot(4,1,2)
plt.plot(widmo[:length//2])
plt.xlim(right = fs/2)
plt.grid(True)
plt.title("Sygnał po transformacie")

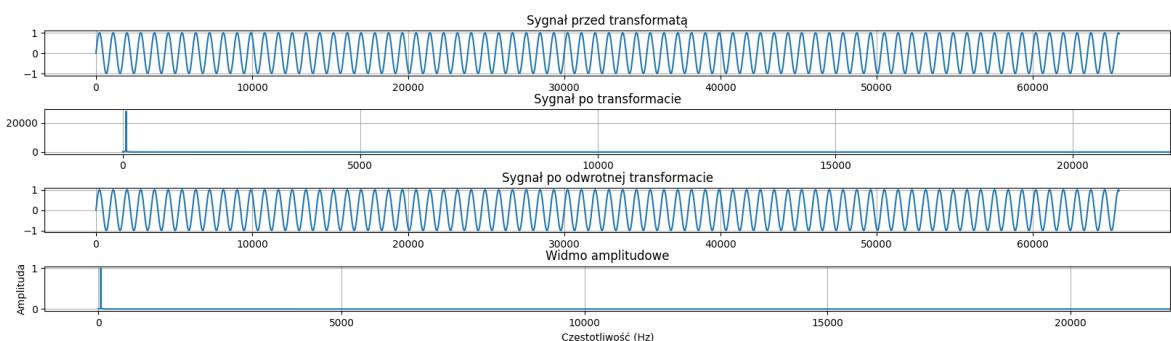
fourier2 = np.fft.ifft(fourier1)

plt.subplot(4,1,3)
plt.plot(fourier2)
plt.grid(True)
plt.title("Sygnał po odwrotnej transformacie")

plt.subplot(4,1,4)
plt.plot(pos_freq, pos_widmo)
plt.xlim(right = fs/2)
plt.grid(True)
plt.title("Widmo amplitudowe")
plt.xlabel('Częstotliwość (Hz)')
plt.ylabel('Amplituda')

plt.subplots_adjust(hspace=0.75)
plt.show()

```



## Zadanie nr2

Przy użyciu biblioteki np.fft tworzona jest transformata fouriera i widmo sygnału.

```

In [ ]: #
plt.figure(figsize=(20,5))
plt.subplot(4,1,1)
plt.plot(mixed_wave)
plt.grid(True)
plt.title("Sygnał przed transformata")

fourier1 = np.fft.fft(mixed_wave)
widmo = np.abs(fourier1)
abs_widmo = widmo / np.max(widmo)
freq = np.fft.fftfreq(len(t), 1/fs)
pos_freq = freq[:len(freq)//2]
pos_widmo = abs_widmo[:len(abs_widmo)//2]

```

```

plt.subplot(4,1,2)
plt.plot(widmo[:length//2])
plt.xlim(right = fs/2)
plt.grid(True)
plt.title("Sygnał po transformacie")

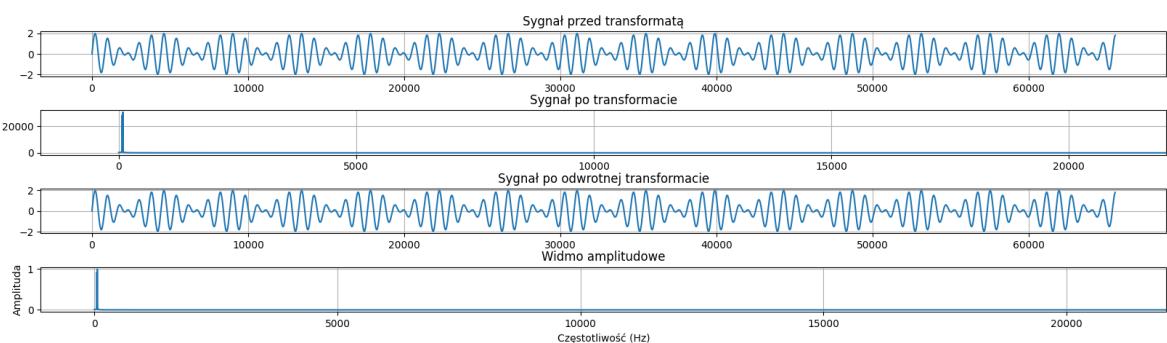
fourier2 = np.fft.ifft(fourier1)

plt.subplot(4,1,3)
plt.plot(fourier2)
plt.grid(True)
plt.title("Sygnał po odwrotnej transformacie")

plt.subplot(4,1,4)
plt.plot(pos_freq,pos_widmo)
plt.xlim(right = fs/2)
plt.grid(True)
plt.title("Widmo amplitudowe")
plt.xlabel('Częstotliwość (Hz)')
plt.ylabel('Amplituda')

plt.subplots_adjust(hspace=0.75)
plt.show()

```



### Zadanie nr3

Zadanie nr3 jest powtórką zadania nr 2 więc kod będzie w zasadzie taki sam (różnicą jest sygnał wykorzystany do w transformacie fouriera).

### Zadanie nr5

Zadanie nr5 zostało rozwiążane odpowiednio w zadaniu nr2 i nr3.