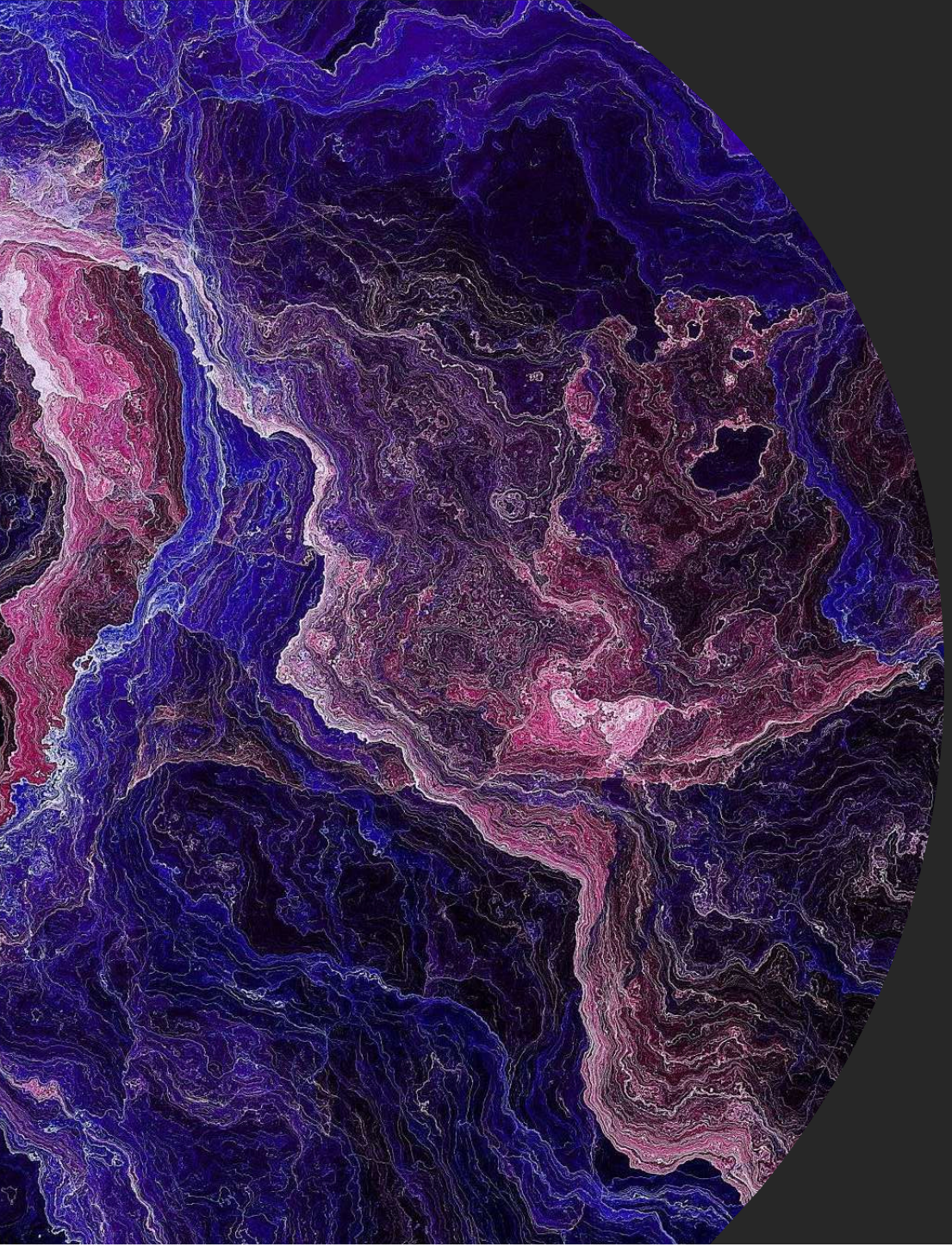
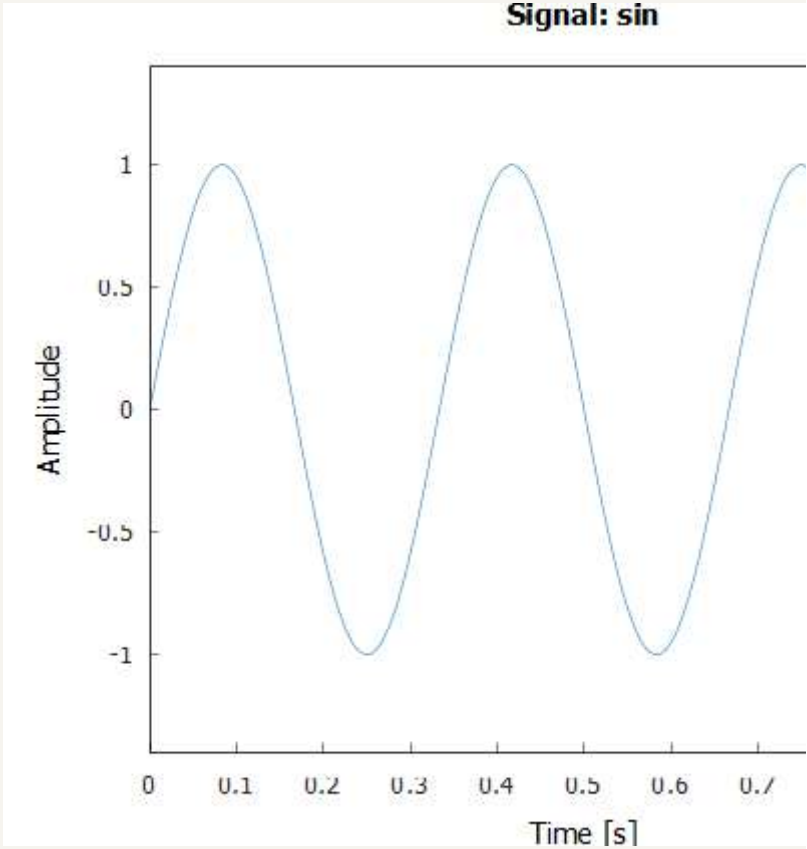


Raport Projekt 3

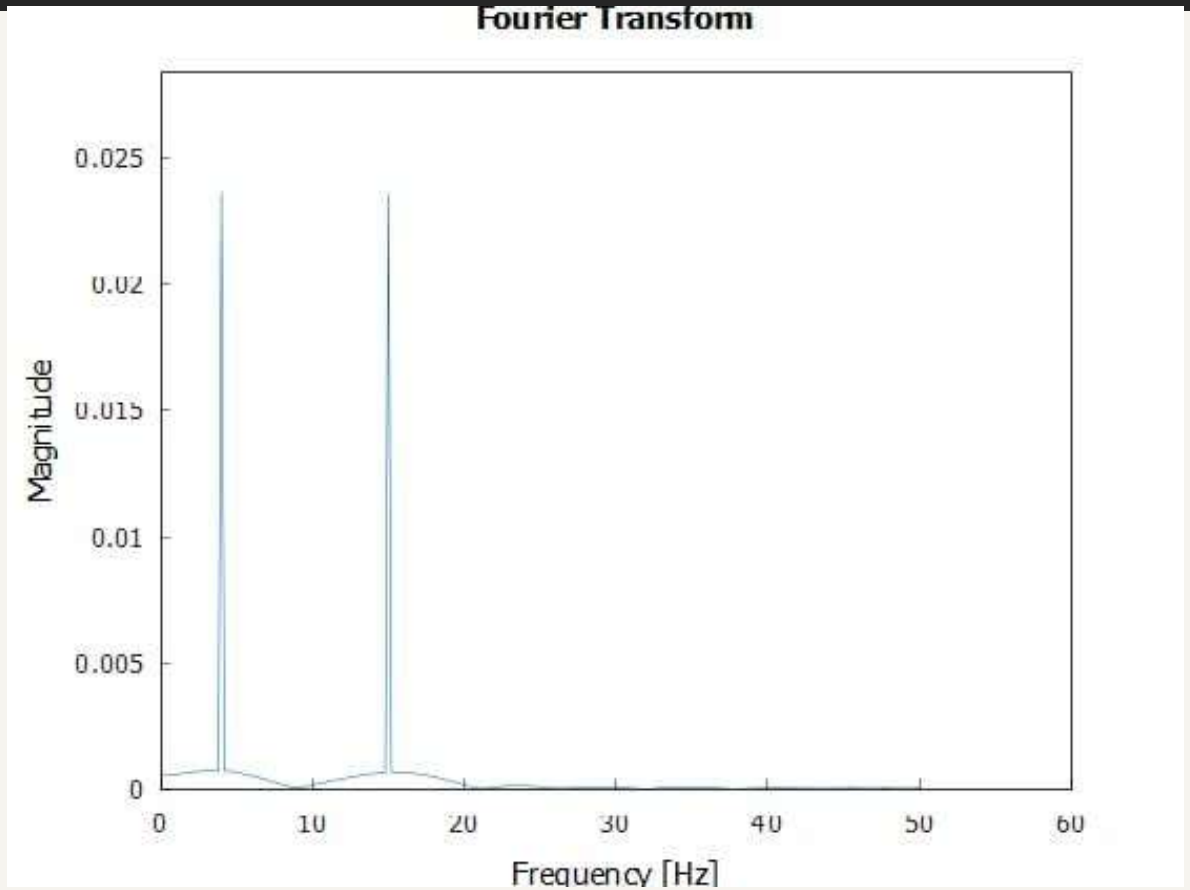
Filip Strawa s203655
Marek Kulma s203260
Grupa 2A

Wizualizacja wykresów 1D

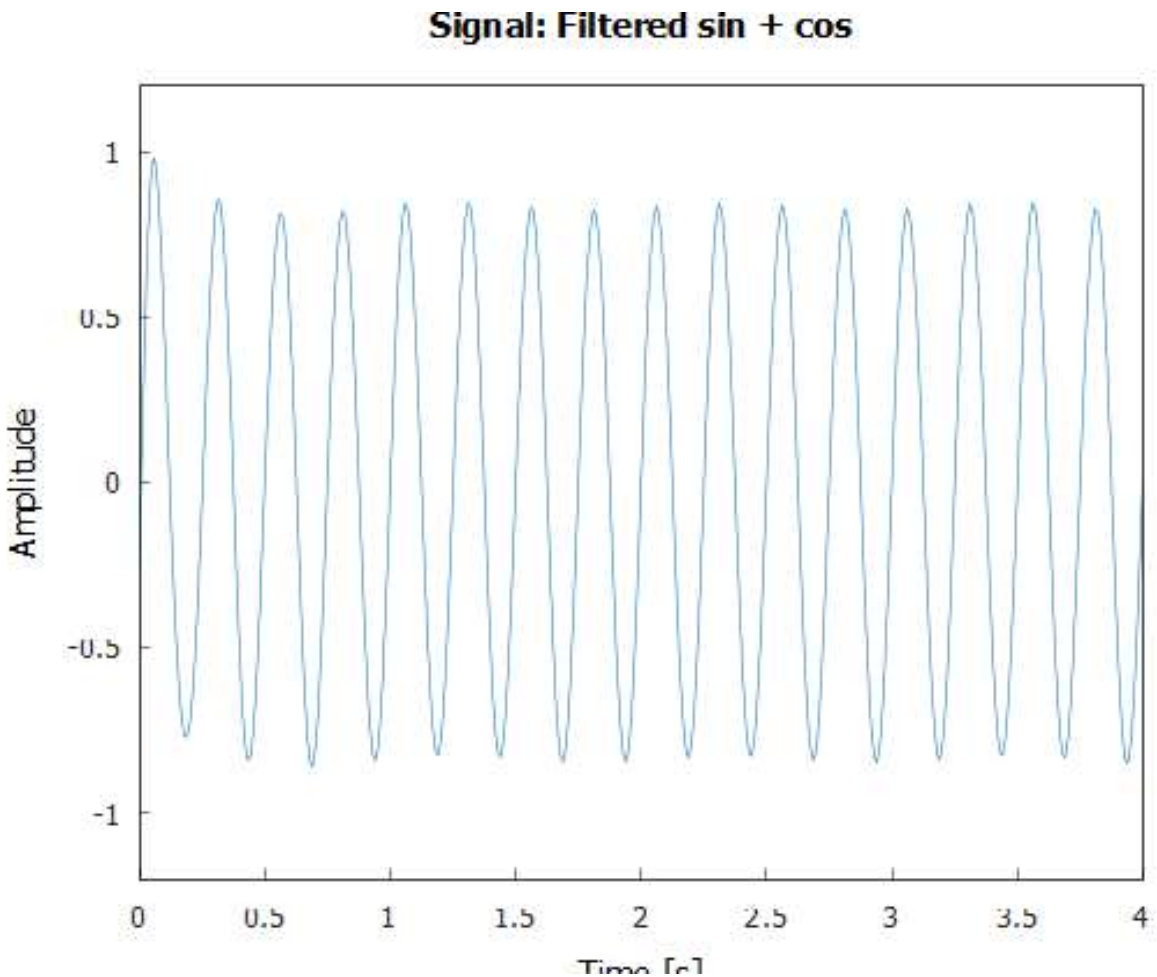
```
Signal filter(Signal signal,double cutoff_hz) {
    Fourier transformata(signal);
     int M  = signal.samples.size();  // liczba próbek
    double fs = N;                   // globalne N = częst. próbkowania [Hz]
    double df = fs / M;              // odstęp między kolejnymi binami

    // 1) DFT



    // 2) zerowanie składowych |f| > cutoff_hz
    for (int k = 0; k < M; ++k) {
        // wyznacz rzeczywistą częstotliwość binu k:
        double freq = k * df;
        if (k >= M/2)             // biny 'użyteczne' w drugiej połowie to od
            freq = (k - M) * df;
        if (std::abs(freq) > cutoff_hz) {
            transformata.X[k] = {0.0, 0.0};
        }
    }Signal filtered_signal("Filtered " + signal.name, transformata);
    return (filtered_signal);
}
```



Wizualizacja DFT




```cpp
void plot_fourier(Fourier fourier) {
    using namespace matplot;

    // upewnia się że jest folder na raport jak nie ma to tworzy
    std::filesystem::path report_path = std::filesystem::current_path().p
    if (!std::filesystem::exists(report_path)) {
        std::filesystem::create_directory(report_path);
    }

    // wektor częstotliwości zależny od próbkowania
    size_t M  = fourier.X.size();  // liczba próbek
    double fs = N;         // częstotliwość próbkowania [Hz]
    double df = fs / M;            // odstęp między kolejnymi momentami c

    // wyświetlić tylko nad kreską
    size_t H = M/2 + 1;            // liczba binów w rfft
    std::vector<double> f(H), magnitude(H);

    for (size_t k = 0; k < H; ++k) {
        // oś częstotliwości w [Hz]
        f[k] = k * df;

        // amplituda – standardowo skalujemy przez 2/M, żeby mieć
        // rzeczywiste wartości z dziedziny czasu
        magnitude[k] = 2.0 * std::abs(fourier.X[k]) / M;

    }
    //rysowanie wykresów
    xlim({0, *std::max_element(f.begin(),f.end())* 1.2});
    ylim({0, *std::max_element(magnitude.begin(), magnitude.end()) * 1.2
    plot(f, magnitude);
    xlabel("Frequency [Hz]");
    ylabel("Magnitude");
    title("Fourier Transform");

    std::string filename = (("../raport/Fourier_transform.png"));
    std::cout << "Saving plot to: " << filename << std::endl;
    save(filename);
    show();
}
```

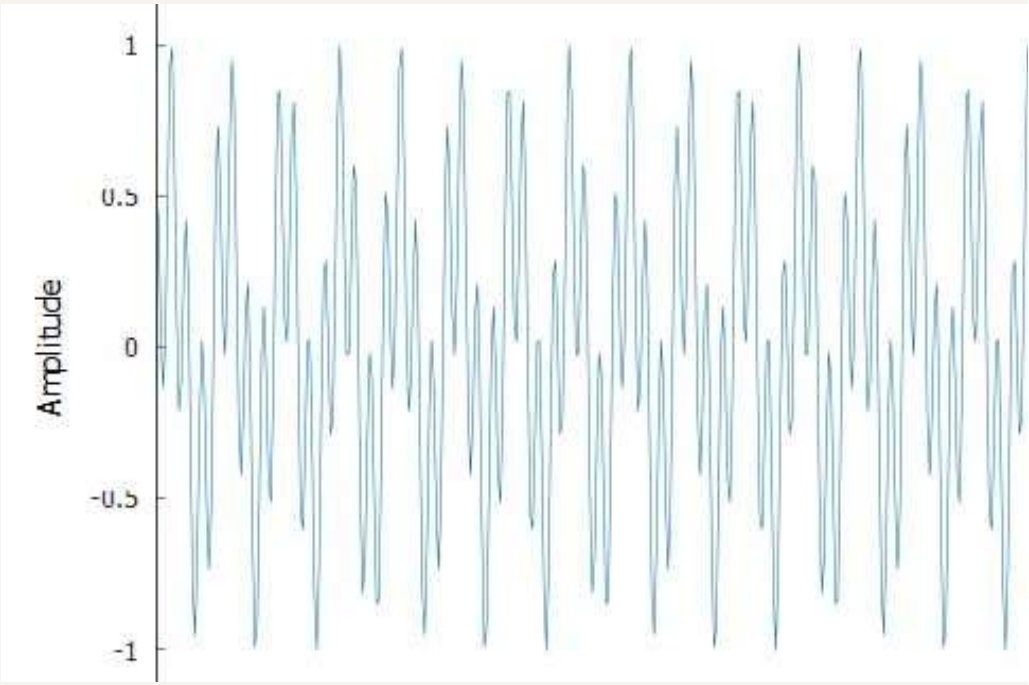


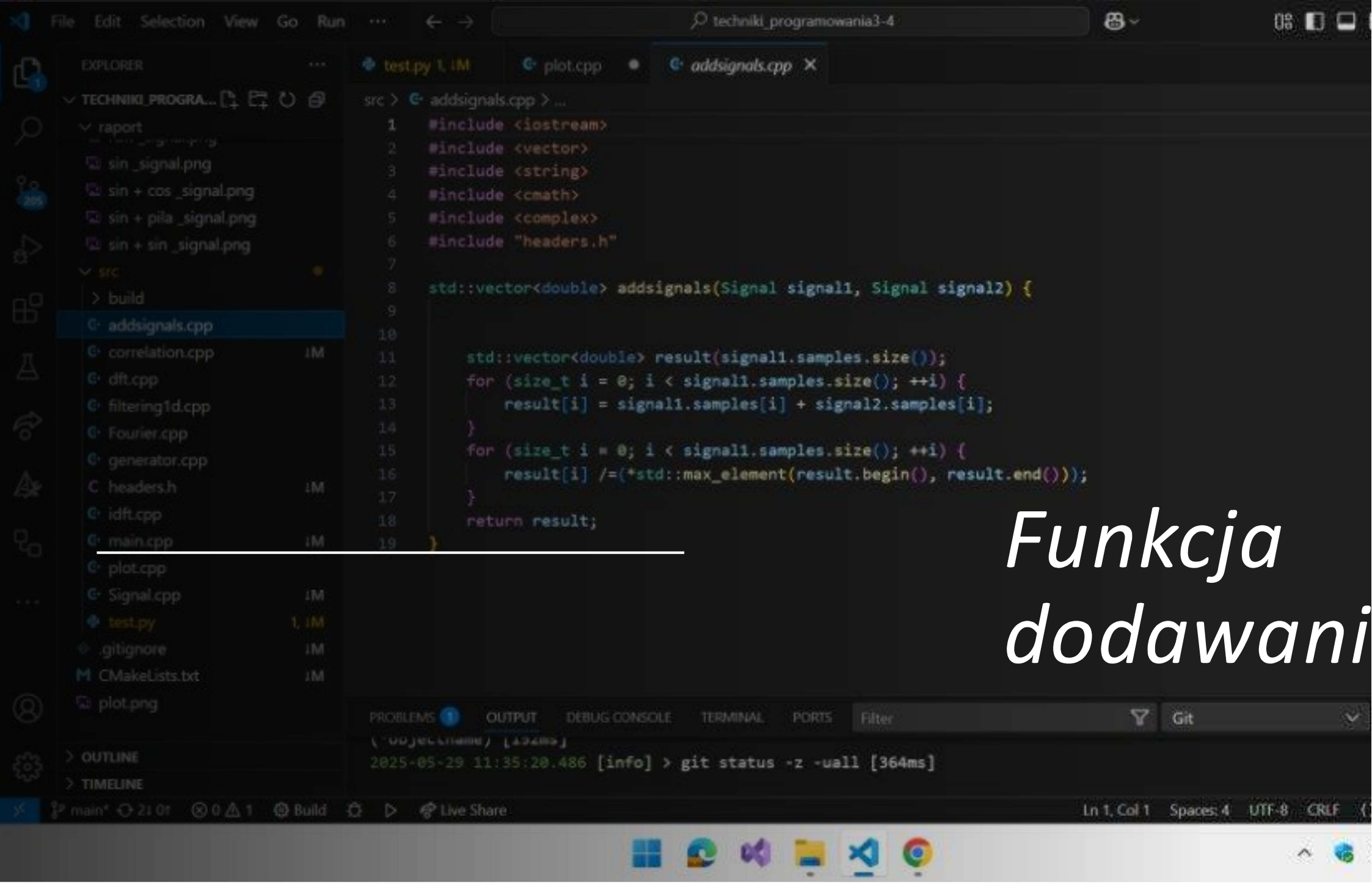

```
#include "headers.h"
Signal filter(Signal signal,double cutoff_hz) {
    Fourier transformata(signal);
    int M  = signal.samples.size();  // liczba próbe
    double fs = N;                   // globalne N =
    double df = fs / M;              // odstęp między

    // 1) DFT


    // 2) zerowanie składowych |f| > cutoff_hz
    for (int k = 0; k < M; ++k) {
        // wyznacz rzeczywistą częstotliwość binu k:
        double freq = k * df;
        if (k >= M/2)                // biny 'użyteczne'
            freq = (k - M) * df;
        if (std::abs(freq) > cutoff_hz) {
            transformata.X[k] = {0.0, 0.0};
        }
    }Signal filtered_signal("Filtered " + signal.name
    return (filtered_signal);
}
```

Filtrowana suma sygnałów dolnoprzepustowa(
6Hz

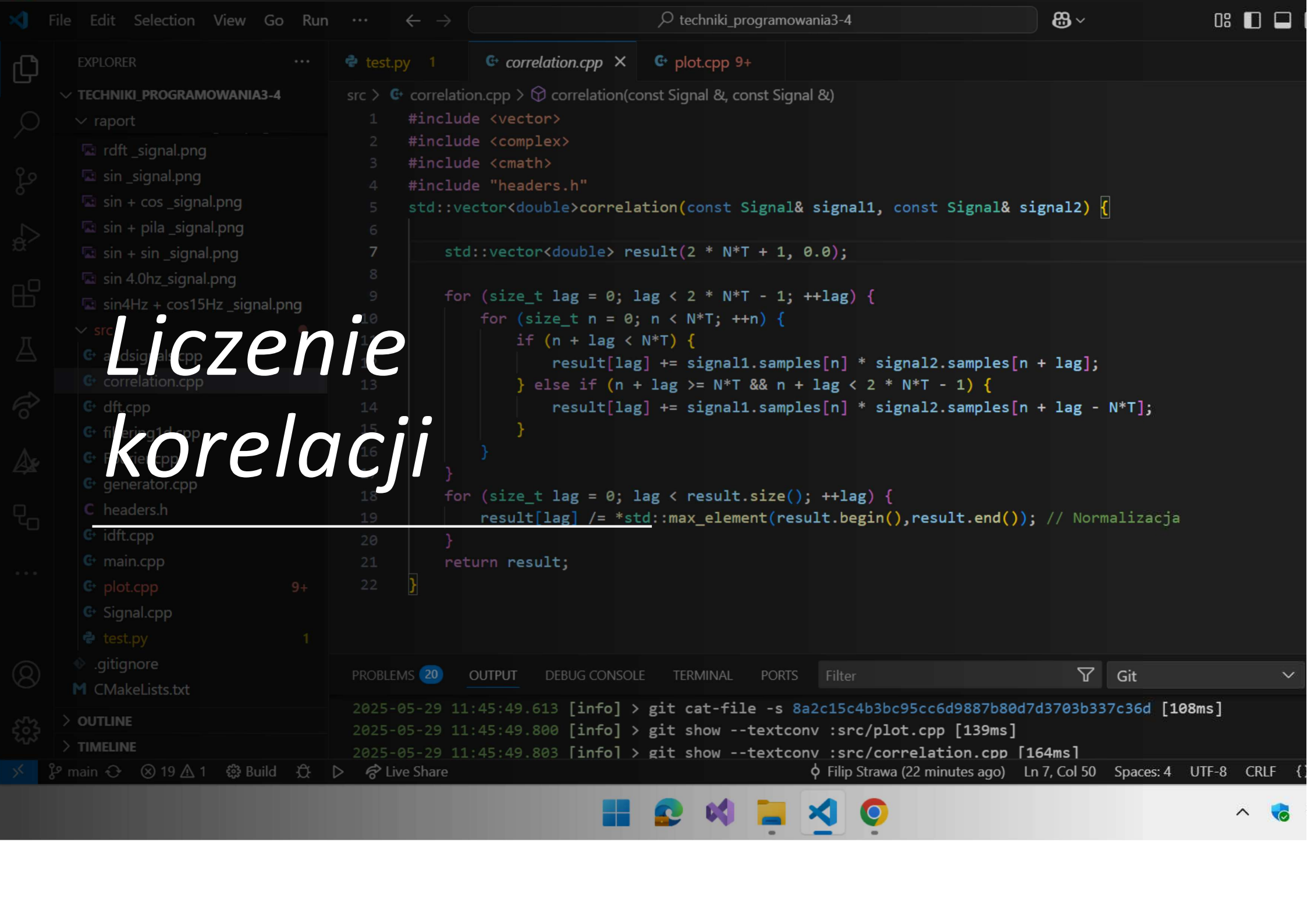
Suma sygnałów bez filtracji

Sin 4Hz I cosinus 15Hz

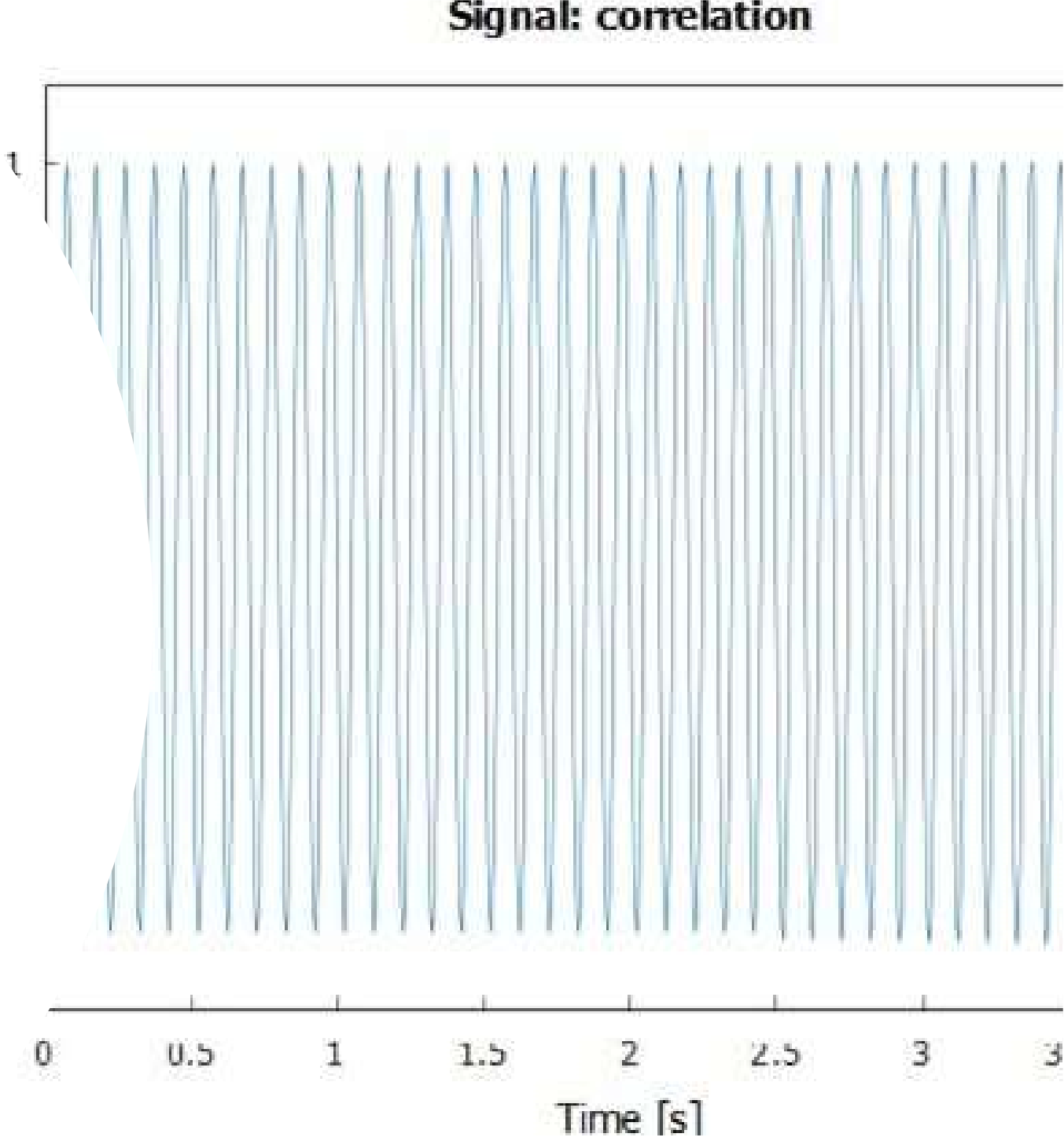



File Edit Selection View Go Run
techniki_programowania3-4
EXPLORER
TECHNIKI_PROGRA...
raport
sin _signal.png
sin + cos _signal.png
sin + piła _signal.png
sin + sin _signal.png
src
build
addsignals.cpp
correlation.cpp
dft.cpp
filtering1d.cpp
Fourier.cpp
generator.cpp
headers.h
idft.cpp
main.cpp
plot.cpp
Signal.cpp
test.py
.gitignore
CMakeLists.txt
plot.png
test.py 1, IM
plot.cpp
addsignals.cpp
src > addsignals.cpp > ...
#include <iostream>
#include <vector>
#include <string>
#include <cmath>
#include <complex>
#include "headers.h"
std::vector<double> addsignals(Signal signal1, Signal signal2) {
std::vector<double> result(signal1.samples.size());
for (size_t i = 0; i < signal1.samples.size(); ++i) {
result[i] = signal1.samples[i] + signal2.samples[i];
}
for (size_t i = 0; i < signal1.samples.size(); ++i) {
result[i] /=(*std::max_element(result.begin(), result.end()));
}
return result;
}
Funkcja dodawani
PROBLEMS 1
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
Filter
Git
2025-05-29 11:35:20.486 [info] > git status -z -uall [364ms]
OUTLINE
TIMELINE
Build
Live Share
Ln 1, Col 1
Spaces: 4
UTF-8
CRLF

Liczenie korelacji

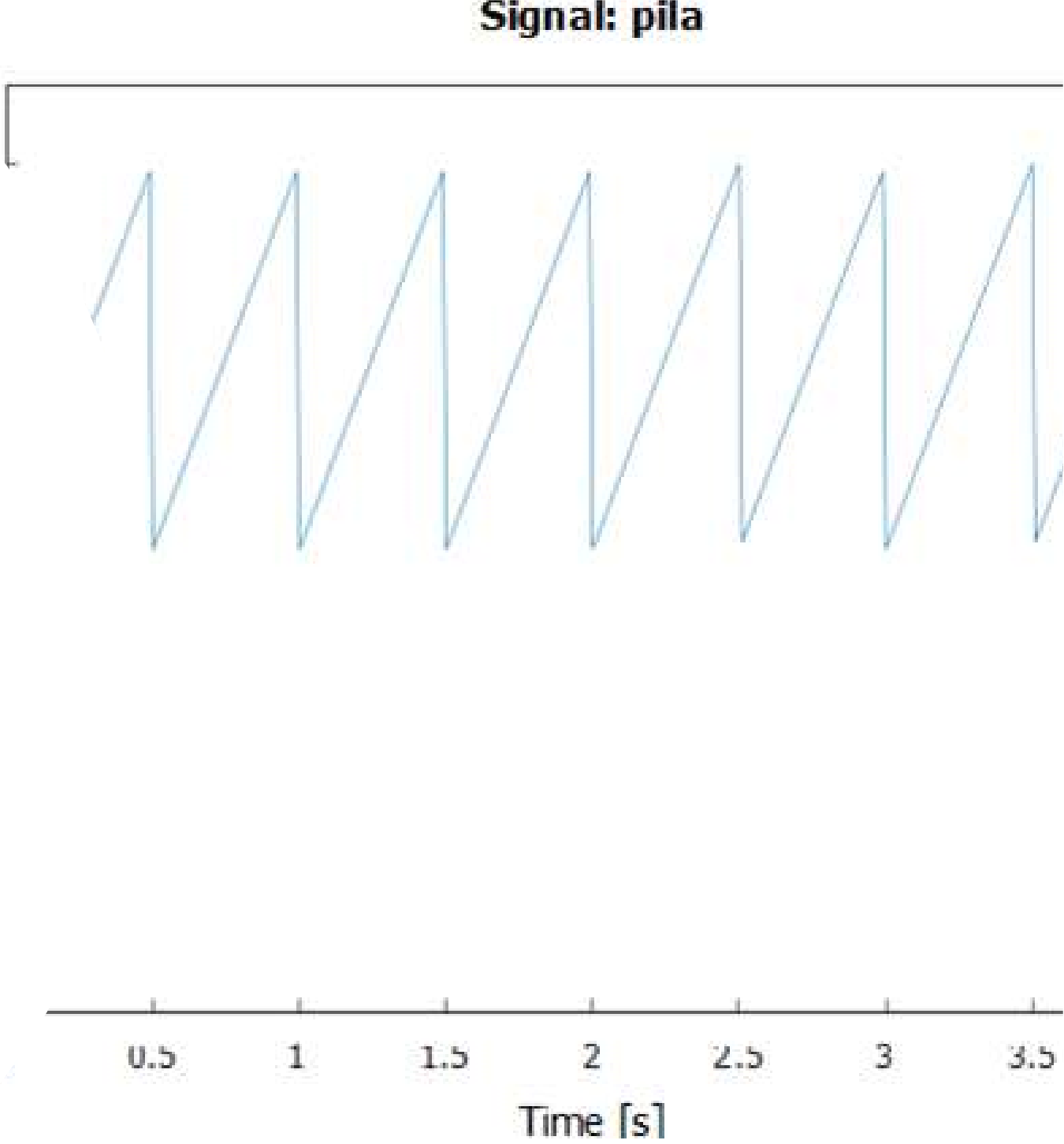


Efekt dla sin i cos oba 5Hz

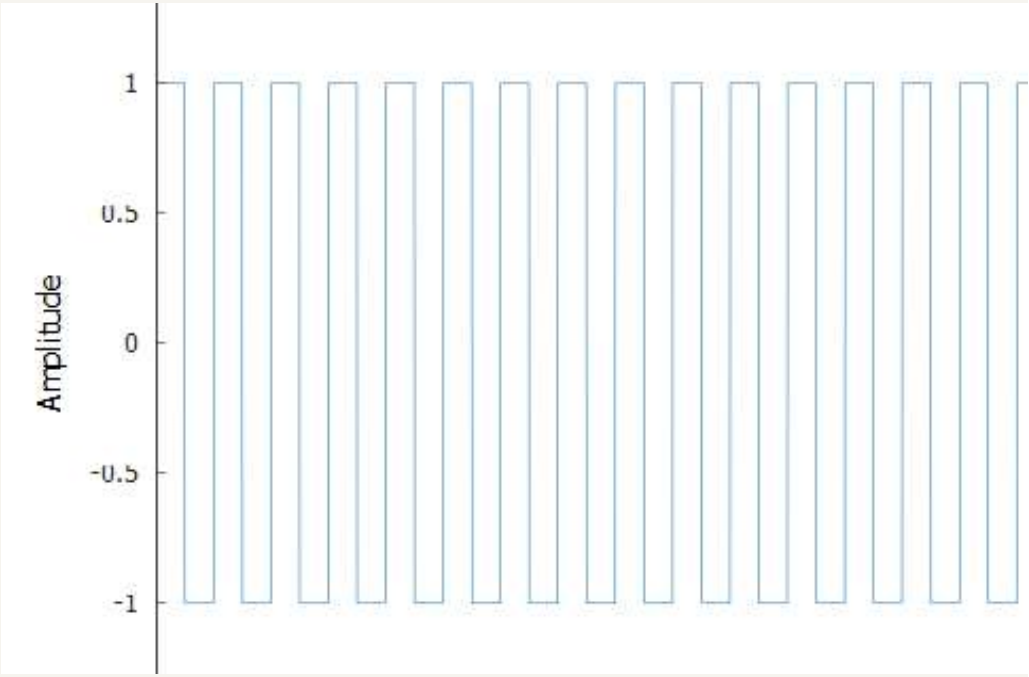


Inne sygnały

Piła 2Hz



Kwadrat 4Hz



Myśl przewodnia

```
wybor = input("podaj wybór (g - generuj drugi sygnał, f - filtruj, add - dodaj dwa sygnały, p - plotuj, pf- plotu
while wybor!="q": #wielkie menu tylko że to pyton
    if wybor == "g":
        freaquency = float(input("podaj częstotliwość sygnału: "))
        faze = float(input("podaj przesunięcie fazowe: "))
        signal_type = input("podaj typ sygnału (sin,cos, kwadrat, pila): ")
        signallist.append(example.Signal(freaquency, faze, signal_type))
    elif wybor=="add":
        signallist.append(example.Signal((signallist[wypiszelementy(signallist)]),signallist[wypiszelementy(signa
    elif wybor == "f":
        cutoff_frequency = float(input("podaj częstotliwość odcięcia: "))
        signallist.append(example.filter((signallist[wypiszelementy(signallist)]),cutoff_frequency))
    elif wybor == "p":
        print("Wybierz sygnał do wyświetlenia:")
        example.plot_signal(signallist[wypiszelementy(signallist)])
    elif wybor == "pf":
        print("wybierz transformację Fouriera do wyświetlenia:")
        example.plot_fourier(fourierlist[wypiszelementy(fourierlist)])
    elif wybor == "dft":
        fourierlist.append(example.Fourier(signallist[wypiszelementy(signallist)]))
    elif wybor == "rdft":
        indekswyboru = wypiszelementy(fourierlist)
        signallist.append(example.Signal( "rdft",fourierlist[indekswyboru]))
        print("Odwrotna transformata Fouriera została dodana do listy sygnałów.")
    elif wybor=="corr":
        signallist.append(example.Signal(example.correlation(signallist[wypiszelementy(signallist)], signallist[
    else:
        print("Nieznany wybór. Spróbuj ponownie.")
    wybor = input("podaj wybór (g - generuj, f - filtruj, p - plotuj, dft - transformata Fouriera, rdft - odwrot
```

- Każdy z sygnałów zapisywany jest do tab w python za pomocą funkcji append oraz odpowiedniego konstruktora lub odpowiedniej funkcji

Inny sygnał