ŁUKASZ STANISZEWSKI

WPROWADZENIE DO MULTIMEDIÓW

SPRAWOZDANIE Z LABORATORIUM 1 – SYGNAŁY I WIDMA



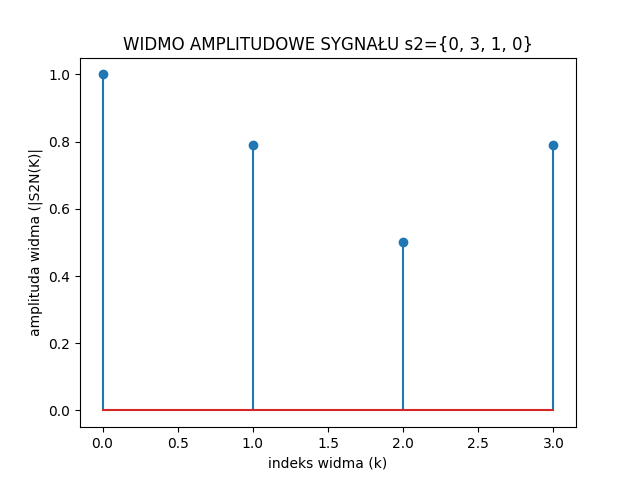
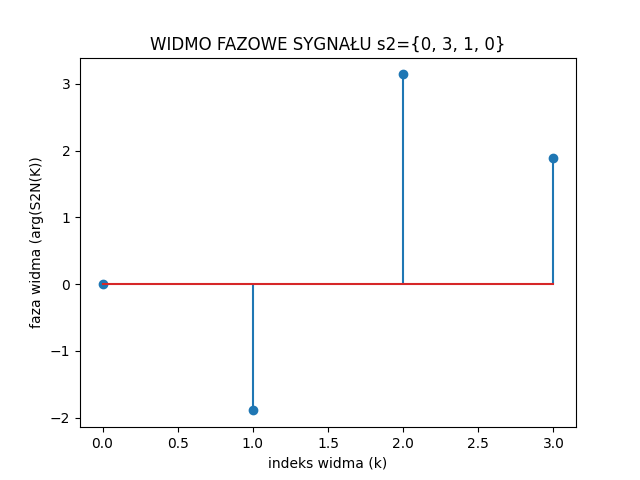
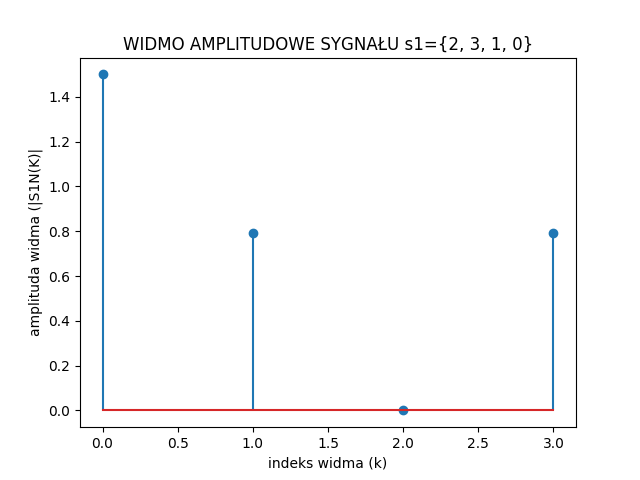
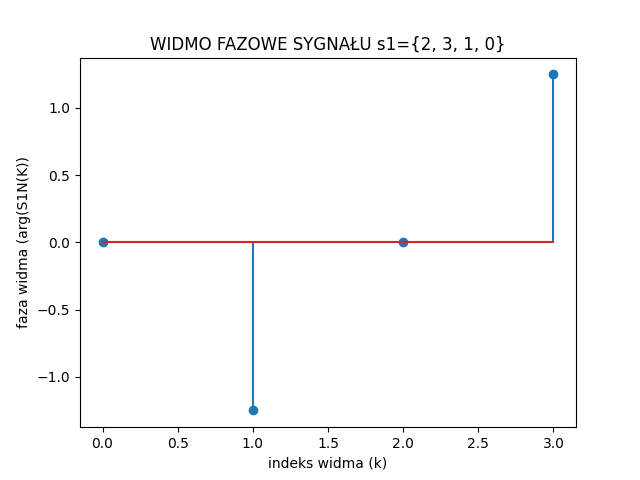
# **Polecenie 1.**

Dane są dwa sygnały o okresie podstawowym : i:

* Dla każdego sygnału wyznaczyć i wykreślić widmo amplitudowe i fazowe, obliczyć moc sygnału i sprawdzić słuszność twierdzenia Parsevala.
* Sprawdzić słuszność twierdzenia o dyskretnej transformacji Fouriera splotu kołowego sygnałów i : wyznaczyć ręcznie splot kołowy sygnałów i , a następnie wyznaczyć ten splot ponownie za pomocą dyskretnej transformacji Fouriera.

UWAGA: ROZWIĄZANIE POLECENIA NR 1 ZNAJDUJE SIĘ W PLIKU **ex1.py**.

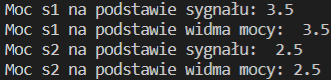
## **WYZNACZENIE WIDM SYGNAŁÓW**

W rozwiązaniu skorzystałem z funkcji **fft()** z modułu **np.fft**, obliczającej szybką transformatę Fouriera. Następnie wyniki podzieliłem przez długość okresu **N**, aby wzór był zgodny z przekształceniem **DFT** prezentowanym na wykładzie. Następnie skorzystałem z funkcji **np.abs()** oraz **np.angle()** w celu otrzymania odpowiednio: widma amplitudowego i widma fazowego sygnałów. W wyniku tego działania otrzymałem 4 następujące wykresy:

UWAGA: należy pamiętać, że każde z powyższych widm jest **4-okresowe**.

## **Sprawdzenie twierdzenia Parsevala**

Twierdzenie Parsevala określa, że moc średnia dla okresowego sygnału czasu dyskretnego może zostać obliczona jako suma widm amplitudowych podniesionych do kwadratu (widm mocy):

Dlatego też w pliku z rozwiązaniem powstały dwie funkcje: **px\_per\_sig\_disc(sig, periodN)** oraz **parseval\_dft(sig)** obliczające odpowiednio lewą i prawą stronę powyższego równania:

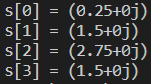
Jak można zauważyć porównując wyniki, otrzymane wyniki się zgadzają co potwierdza słuszność twierdzenia Parsevala.

## **Sprawdzenie twierdzenia o DFT splotu kołowego sygnałów**

Należy udowodnić, że:

W tym celu najpierw obliczona została ręcznie wartość z użyciem wzoru :

* dla n = 0:
* dla n = 1**:**
* dla n = 2:
* dla n = 3:

Następnie sploty podzielono przez długość okresu **N**, a wynikiem jest zbiór: . Wartość tą należy teraz porównać z przemnożonymi przez siebie widmami **S1(k)** i **S2(k)**, następnie iloczyn ten przekształcić z użyciem **np.fft.ifft()** oraz przemnożyć wynik przez **N** (dla uzgodnienia wzoru na **DFT**). Otrzymane wyniki:

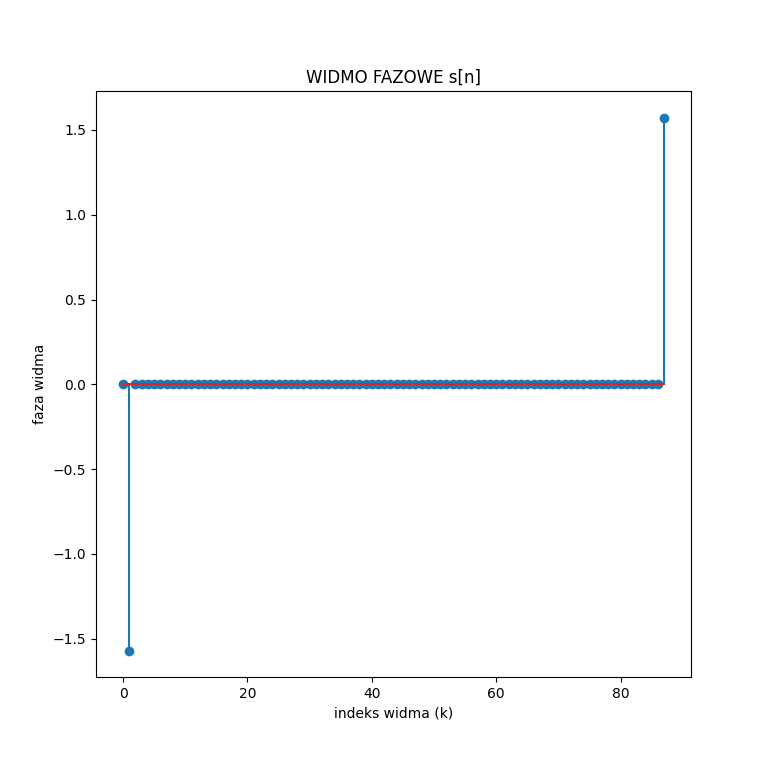
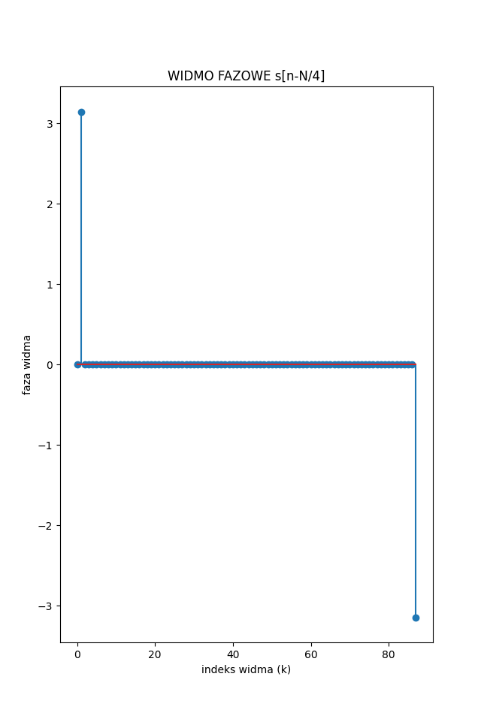
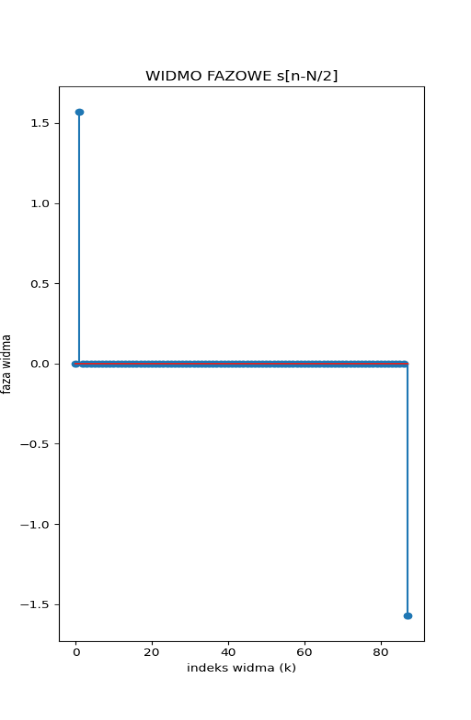
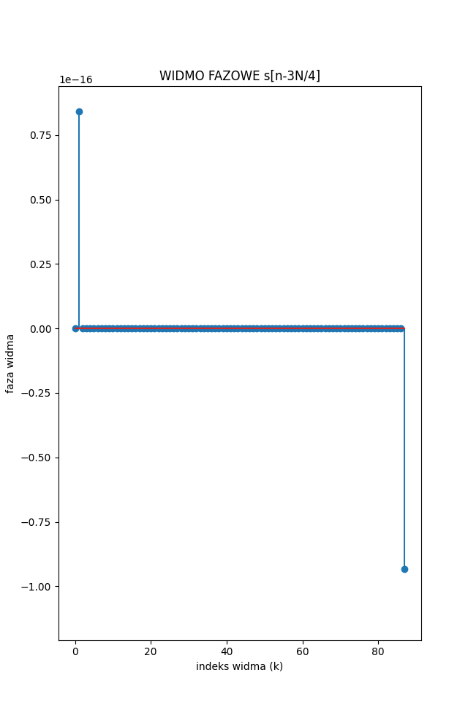
Wyniki te (porównane z obliczeniami) potwierdzają słuszność powyższego twierdzenia.

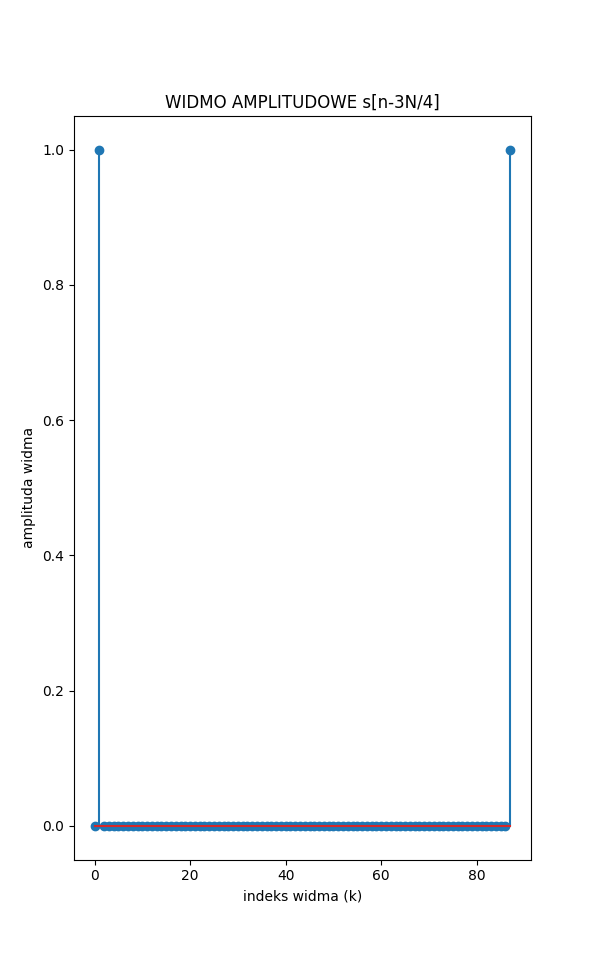
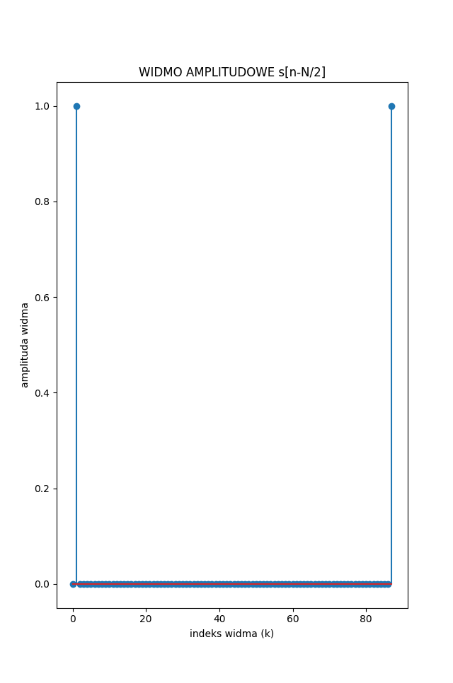
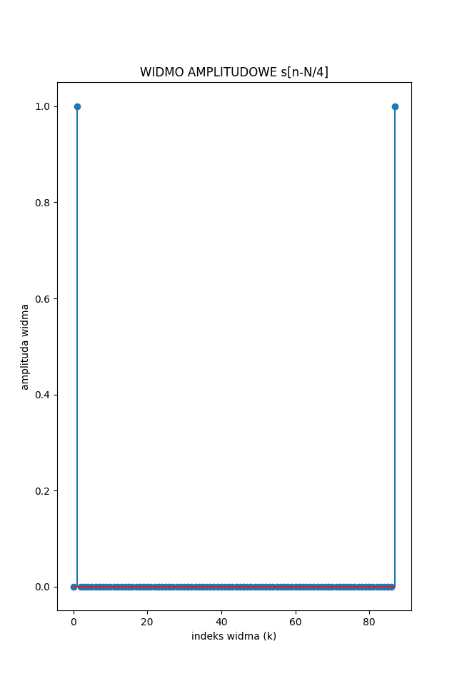
# **POLECENIE II.**

Zbadać wpływ przesunięcia w czasie na postać widma amplitudowego i widma fazowego dyskretnego sygnału harmonicznego o amplitudzie i okresie podstawowym . W tym celu dla każdej wartości wykreślić widmo amplitudowe i fazowe przesuniętego sygnału **.** Skomentować otrzymane wyniki.

UWAGA: ROZWIĄZANIE POLECENIA NR 2 ZNAJDUJE SIĘ W PLIKU **ex2.py**.

## **Implementacja**

Obraz zawierający kwadrat

Opis wygenerowany automatycznieW rozwiązaniu została zaimplementowana funkcja **signal**() wyliczająca wartość funkcji **.** Za jej pomocą zostały policzone wartości przesuniętego sygnału **,** a następnie przetransformatowane z użyciem funkcji **np.fft.fft()** wraz z podzieleniem przez długość okresu **N**. W wyniku otrzymane zostały następujące wykresy:

## Wnioski

* Na podstawie otrzymanych wykresów amplitud widm można zauważyć, że **przesunięcie w czasie zupełnie nie wpływa na charakterystykę widma amplitudowego danego sygnału**.
* Zmiany można zauważyć zaś w przypadku widm fazowych, mianowicie **zmieniają się wartości faz w zależności od przesunięcia**. Jako, że zbiór wartości widma fazowego to tak więc można wywnioskować, że każde z kolejnych przesunięć zmniejsza – w przypadku takich okresów - wartość fazy o kolejno dla przesunięć dla dwóch skrajnych prążków (u pozostałych by również było to widoczne, jednak w programie zostało zastosowane zerowanie wartości w wektorze transformat dla tych elementów, których amplituda była bardzo bliska 0 w celu **odszumienia widm fazowych**).
* Wnioski te można wysunąć również wzorując się na wzorze:

**,**

widać tu mianowicie, że przesunięcie powoduje zmianę argumentu liczby zespolonej, a moduł się nie zmienia, co zgadza się z wysuniętymi wnioskami na podstawie wykresów.

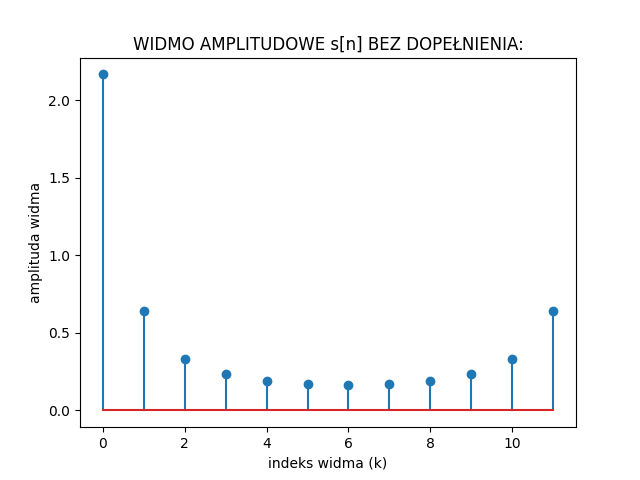
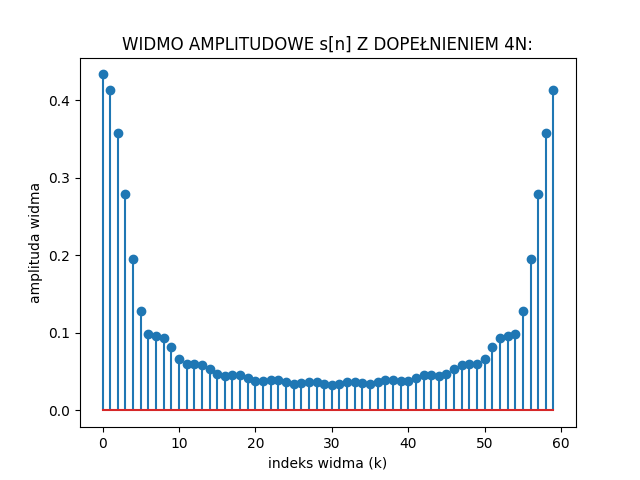
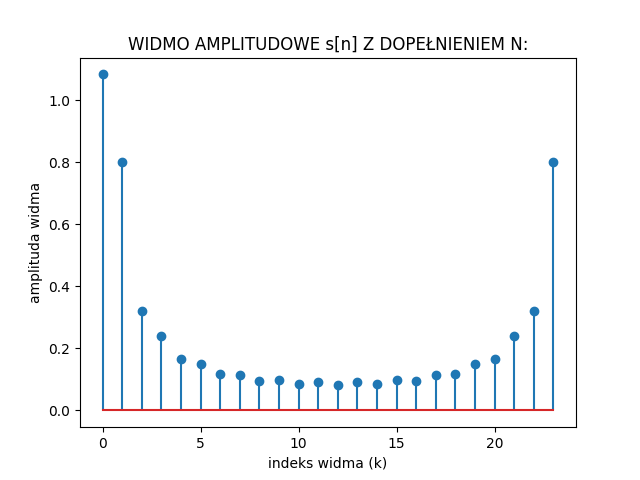
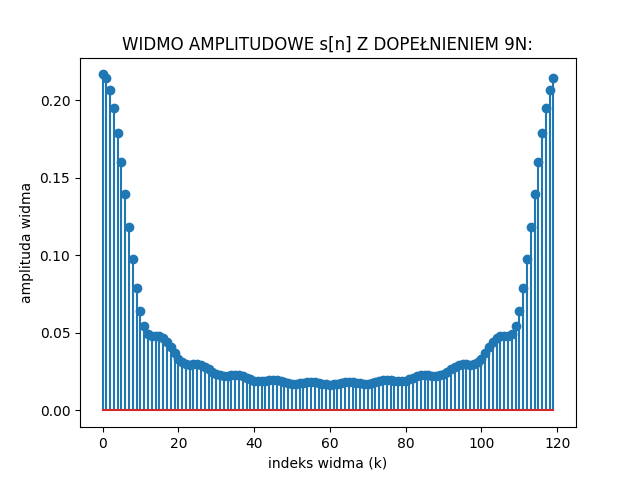
# **Polecenie III.**

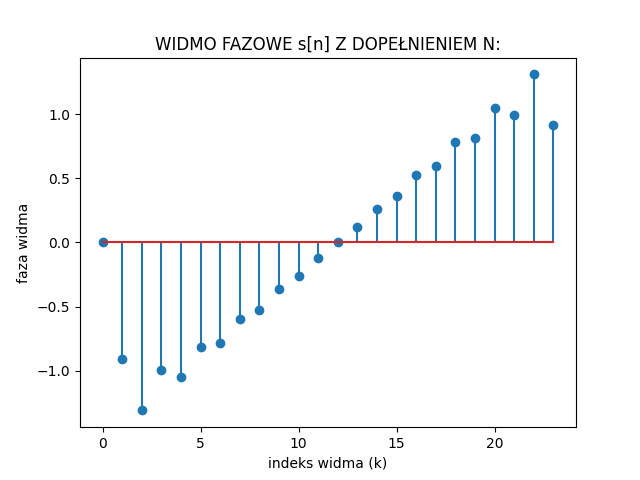
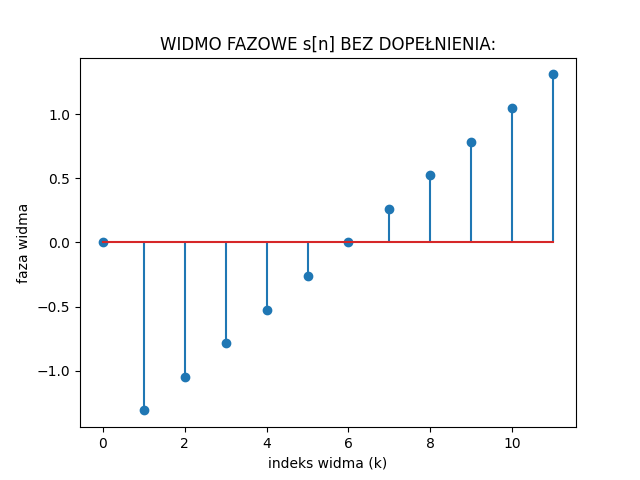
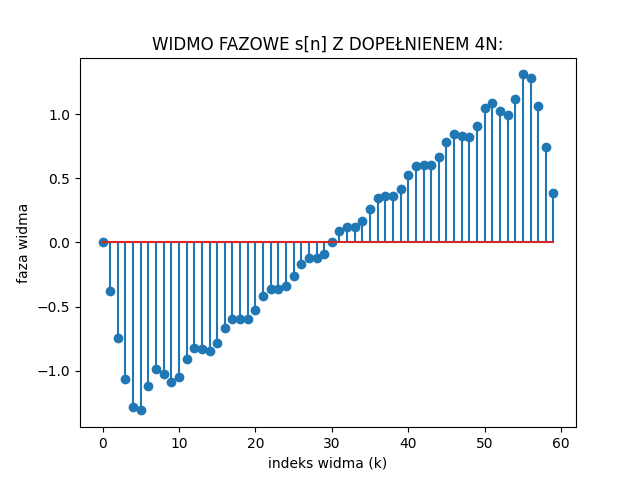
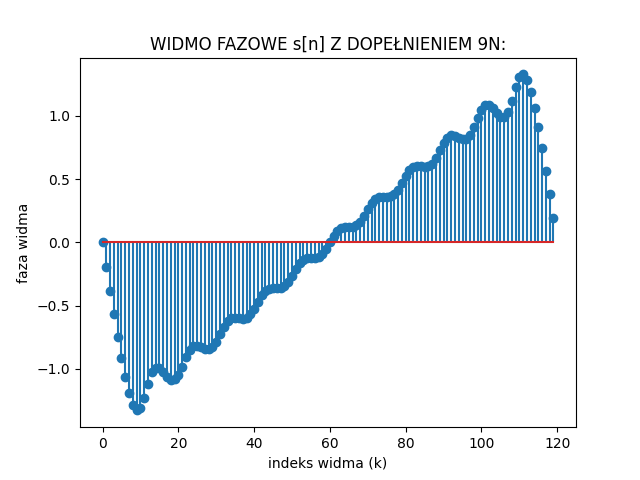
Zbadać wpływ dopełnienia zerami na postać widma amplitudowego i widma fazowego dyskretnego sygnału o amplitudzie i okresie podstawowym . W tym celu dla każdej wartości wykreślić widmo amplitudowe i fazowe sygnału dopełnionego zerami. Skomentować otrzymane wyniki.

UWAGA: ROZWIĄZANIE POLECENIA ZNAJDUJE SIĘ W PLIKU **ex3**.**py**.

## **Implementacja**:

W rozwiązaniu została zaimplementowana funkcja **signal()** wyliczająca wartość funkcji oraz funkcja **normalize()**, która „zeruje” każdy składnik wektora transformat, dla którego amplituda jest bardzo bliska 0 (w celu odszumienia widma fazowego). Funkcja **signal()** została wywołana dla 12 pierwszych próbek, tworząc wektor próbek, który następnie w kolejnych etapach jest dopełniany odpowiednią liczbą zer. Następnie została zastosowana transformata **fft()** z biblioteki **np.fft**, wyniki zostały podzielone przez nowe długości okresów, a wektory transformat znormalizowane wg amplitudy widma. Końcowym wynikiem jest zbiór wykresów opisujący widma amplitud i faz dla odpowiednio dopełnionych zerami sygnałów.





## **Wnioski**

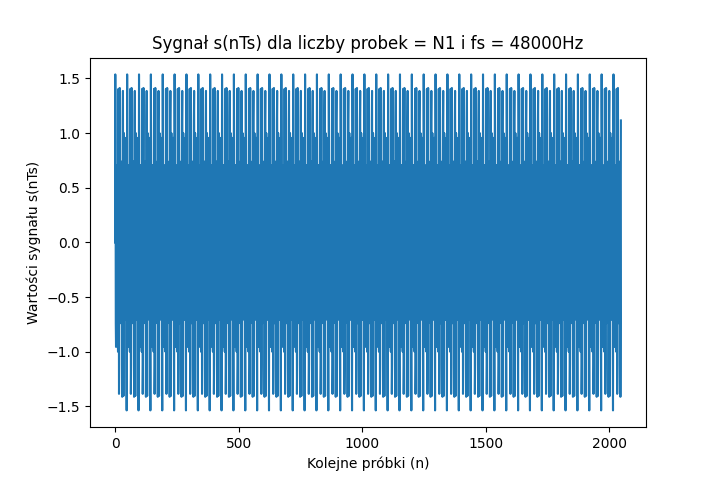
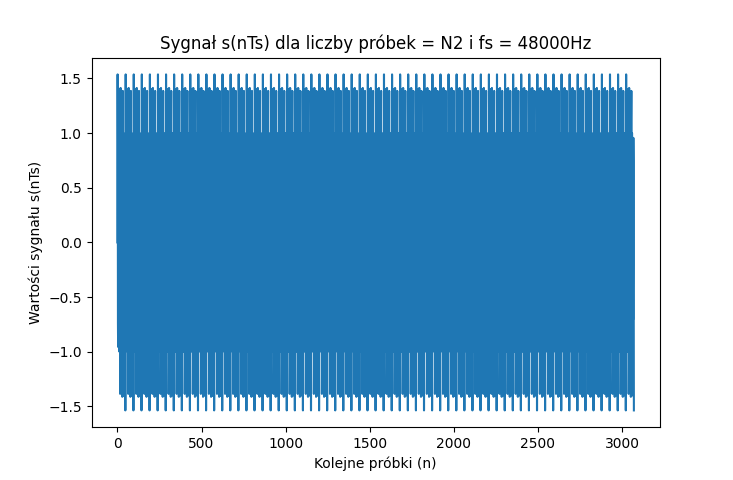
* W tym przypadku zastosowanie dopełnienia zerami ma **znaczący wpływ** zarówno na postać **widma fazowego** jak i **widma amplitudowego** dyskretnego sygnału.
* Można zauważyć, że dopełnienie zerami znacząco **zwiększyło okresy** zarówno sygnału jak i jego widm (kolejno do N=12, 24, 60, 120).
* Na podstawie wykresów można zauważyć, że uzupełnienie dodatkowymi zerami sygnału **zmniejsza różnicę pomiędzy kolejnymi wartościami widma** i **lepiej uwypukla szczegóły widma** (kształt widma się nie zmienia w okresie, rozszerza się on tylko po osi poziomej).
* Własność ta na pewno mogłaby być wykorzystana do rozszerzania liczby próbek sygnału dyskretnego do potęgi liczby 2, aby móc zastosować go do obliczeń algorytmu FFT.
* W tych przypadkach doszło również do zmniejszenia amplitud widma, jest to jednak związane ze zwiększeniem okresów kolejnych sygnałów i zastosowanie dzielenia przez ich długość przy transformacie.

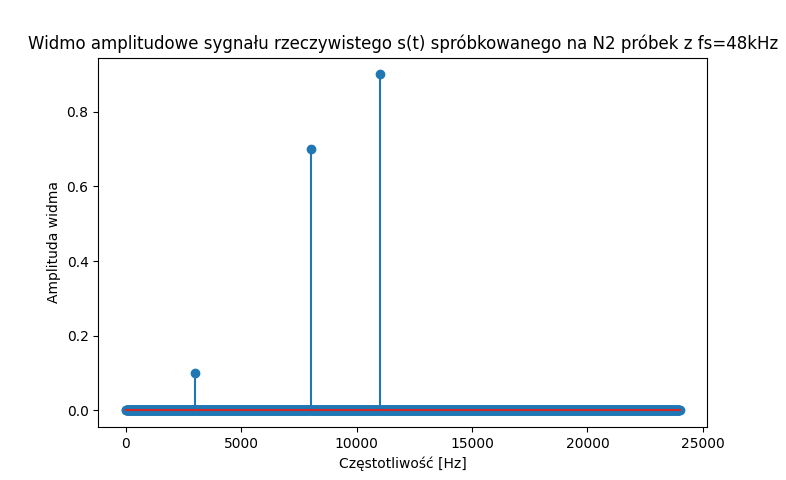
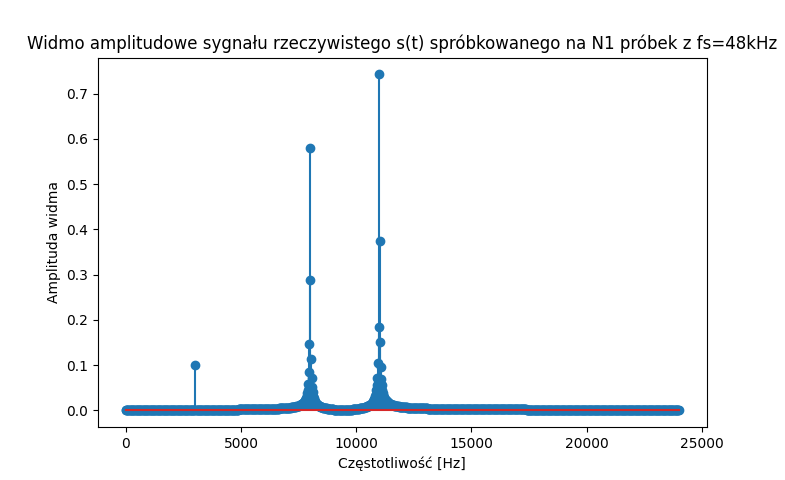
# **POLECENIE IV.**

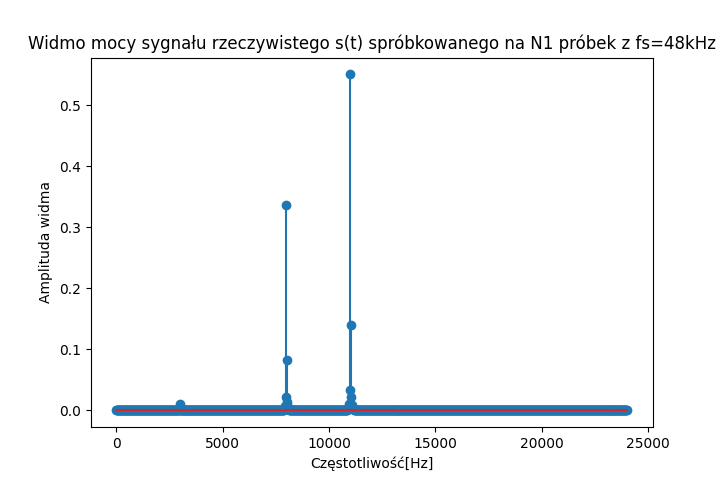
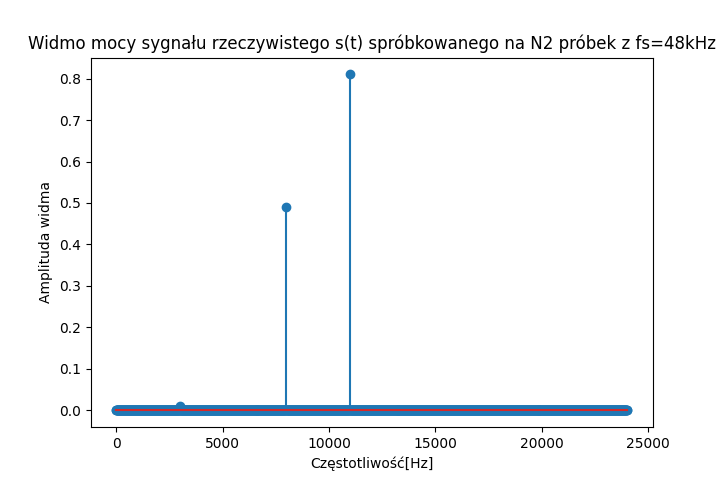
Dany jest sygnał rzeczywisty , gdzie , , , , , . Przy założeniu, że liczba próbek sygnału wynosi , przedstawić wykres **widmowej gęstości mocy sygnału** . Czy dla podanej liczby próbek mamy do czynienia ze **zjawiskiem przecieku widma**? Czy sytuacja uległaby zmianie dla liczby próbek ? Odpowiedź uzasadnić.

UWAGA: rozwiązanie zadania znajduje się w pliku **ex4.py.**

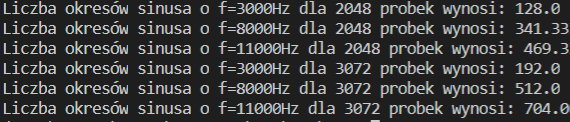
## **Implementacja**

W rozwiązaniu została zaimplementowana funkcja **signal()** zwracająca wartość sygnału **s(t).** Dodatkowo powstała funkcja **perform\_example()**, w której zostały wykonane niezbędne badania dla liczby próbek wynoszącej kolejno **N1** oraz **N2**. W celu wyznaczenia spróbkowanego sygnału **zbiór indeksów** próbek został **podzielony** przez **częstotliwość próbkowania**, a także wyznaczony został **wektor wartości sygnałów** dla kolejnych próbek. Następnie zostały narysowane **wykresy reprezentujące sygnały spróbkowane**:

Następnie użyto funkcji **rfft**() oraz **podzielono wynik przez połowę liczby próbek** (widmo jest N-okresowe, jednak dla funkcji **rfft**() obserwowana jest tylko połowa widm, odrzucamy kopie będące lustrzanymi odbiciami) w celu wyznaczenia **widma rzeczywistego**. Następnie wyznaczone zostały **widma amplitudowe** sygnałów. Dodatkowo koniecznym było **przeskalowanie osi na oś częstotliwości**. Gwarantuje to funkcja **rfftfreq**() zwracająca zbiór częstotliwości dla zadanej liczby próbek i odległości w czasie między próbkami. Następnie zostały narysowane **wykresy widm amplitudowych** dla sygnałów:

Dodatkowo zostały wykreślone **wykresy widm gęstości mocy** (poprzez podniesienie widm amplitudowych do kwadratu):

## **Wnioski**:

* Na podstawie wykresów można zauważyć, że widma – w przypadku liczby próbek = - zgadzają się z sygnałem rzeczywistym **s(t)** – na odpowiednich częstotliwościach, będących częstotliwościami sinusów składających się na sygnał, występują prążki widmowe, których amplitudy są równe amplitudom tych składowych sygnału, tzn. dla częstotliwości **3kHz**, **8kHz** oraz **11kHz** wartości prążków widmowych, zgodnie z sygnałem **s(t)**, wynoszą kolejno **0.1**, **0.7** oraz **0.9**.
* Wartości tych prążków nie są jednak idealne w przypadku liczby próbek = . Można zauważyć, sugerując się widmem mocy, że moc zawarta w prążkach rzeczywistych dla sygnału „**rozlała** **się**” na prążki sąsiadujące z nimi. Efekt ten, zwany „**przeciekiem** **widma**”, pojawił się tylko w przypadku próbkowania dla liczby próbek = , co spowodowało „nieidealną” amplitudę tych prążków.
* Przeciek widma jest efektem ucięcia okresów składowych sygnału analogowego przez blok N próbek. Wystąpienie przecieku widma w przypadku 1. i jego brak w przypadku 2. można więc udowodnić odpowiednimi obliczeniami. Przeciek widma nie będzie występował w przypadku tego próbkowania, gdzie czas trwania próbkowanego sygnału będzie wielokrotnościami czasu trwania okresów składowych tego sygnału. Tak więc wystarczy podzielić iloraz liczby próbek i częstotliwości próbkowania przez okresy sinusów składających się na sygnał, co realizuje funkcja **periods()**, jej wyniki wskazują na słuszność powyższych punktów (brak liczb całkowitych dla ):