Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Інституту атомної та теплової енергетики

Кафедра цифрових технологій в енергетиці

**Проектування та аналіз обчислювальних алгоритмів**

ЗВІТ

до лабораторної роботи № 1,2

«Цифрова фільтрація вхідних даних»

Дата «14» жовтеня 2023 Виконав: студент *2* курсу

гр. *ТР-24*

Хоменко Олег Святославович

*(П.І.Б.)*

Оцінка «\_\_\_\_\_\_\_\_\_» Перевірив: доцент

Дата «\_\_\_» \_\_\_\_\_\_\_ 2023 Донець Андрій Георгійович

*(П.І.Б., підпис)*

Київ – 2023

**Завдання:**

1. Розробити застосунок для цифрової фільтрації вхідних даних.

2. Вхідні дані x(i) (тестовий сигнал!) необхідно попередньо розрахувати та записати в файл filename.tst (базові параметри: Т = 1 сек, N = 1000, Δt = 0.001).

3. Для цифрової обробки вхідних даних застосувати низькочастотний та високочастотний фільтри 1-го та 2-го порядку. Записати вихідні дані y(i) у файли filename.dL1, filename.dL2, filename.dh3 та filename.dh4.

4. Для розрахунку тестового сигналу (файл filename.tst) використати дані, що сформовані на базі наступної математичної моделі:

x(i) = A1 \* sin(w1 \* Δt + φ0) + A2 \* sin(w2 \* Δt + φ0) + A3 \* sin(w3 \* Δt + φ0),

де wk = 2 \* π \* fk;

fk = 50, 60, 400 Гц;

Аk = 220, 110, 36 В.

5. Передбачити можливість модифікації вхідних даних для формування тестового сигналу.

6. У якості вхідного сигналу для проведення розрахунків (робочого, не тестового!) використати файл 001.dat з додатку.

7. Результати розрахунків відобразити у вигляді кольорових графіків A = f(t).

**Теоретичні відомості**

Математична модель сигналу

Сигнал x(i), який генерується для тестування і фільтрації, визначається наступною математичною моделлю:

x(i) = A1 \* sin(w1 \* Δt + φ0) + A2 \* sin(w2 \* Δt + φ0) + A3 \* sin(w3 \* Δt + φ0),

де:

* wk = 2 \* π \* fk - кругова частота для сигналу з частотою fk,
* fk = 50, 60, 400 Гц - частоти компонентів сигналу,
* Ak = 220, 110, 36 В - амплітуди відповідних компонентів,
* Δt - крок дискретизації.

Алгоритм роботи програми

Користувач обирає тип сигналу: тестовий або робочий.

Якщо обрано тестовий сигнал:

Генерується сигнал x(i) з використанням заданих параметрів (частоти і амплітуди).

* Зберігається сигнал x(i) у файл filename.tst.
* Сигнал x(i) піддається фільтрації за допомогою низькочастотних і високочастотних фільтрів першого і другого порядку.
* Результати фільтрації записуються у файли filename.dL1, filename.dL2, filename.dh3 і filename.dh4.
* Відображається графік вхідного та фільтрованого сигналів.

Якщо обрано робочий сигнал:

* Зчитується сигнал із файлу 001.dat.
* Відображається графік робочого сигналу.



Використані функції

generate\_test\_signal(T, N, dt, frequencies, amplitudes): Функція для генерації тестового сигналу на основі заданих параметрів. Використовується для створення та збереження сигналу x(i) у файлі filename.tst.

filter\_signal(signal, lowcut, highcut, fs, order=2): Функція для фільтрації сигналу. Використовується для фільтрації сигналу x(i) з заданими параметрами і запису результатів у файли filename.dL1, filename.dL2, filename.dh3 і filename.dh4.

parse\_sequence(filepath): Функція для зчитування даних з файлу 001.dat, які представлені як послідовність 16-бітних цілих чисел. Дані зчитуються та відображаються у вигляді графіка для робочого сигналу.

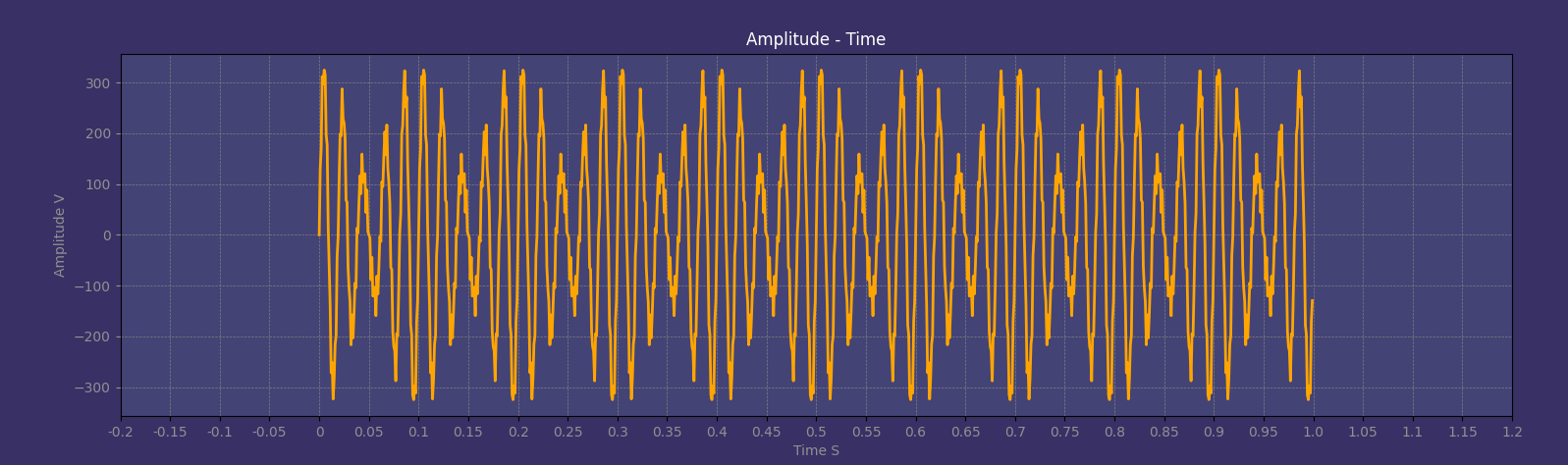
Загальний порядок роботи програми

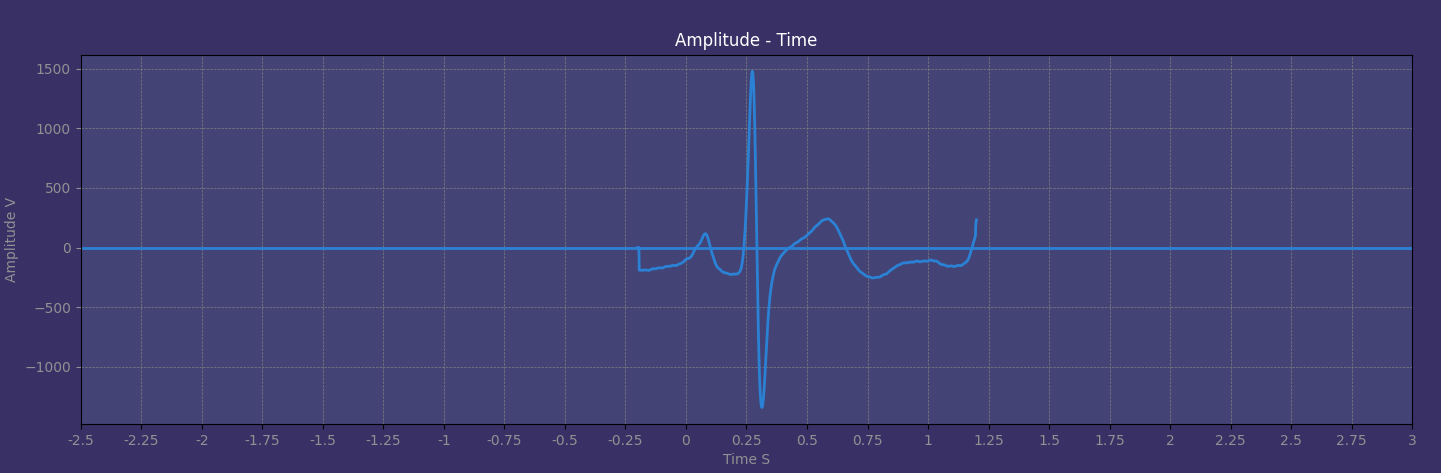
Користувач обирає тип сигналу (тестовий або робочий).

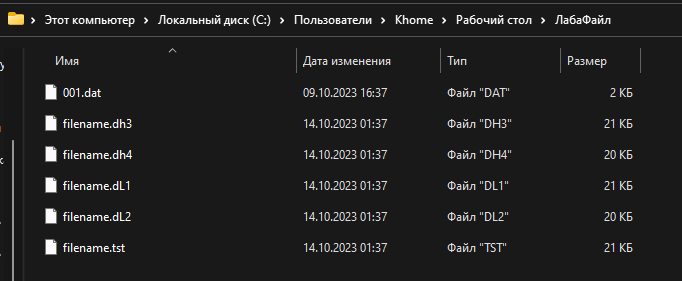
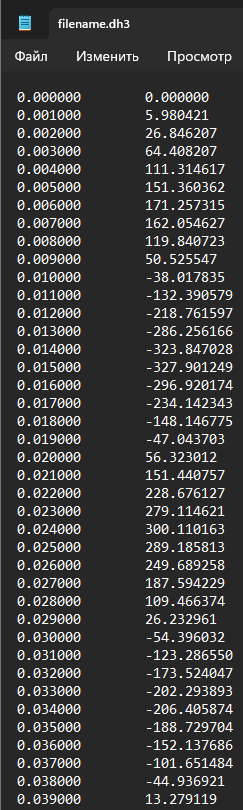
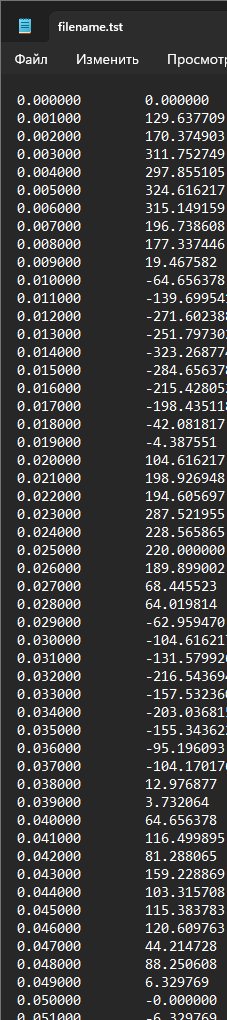
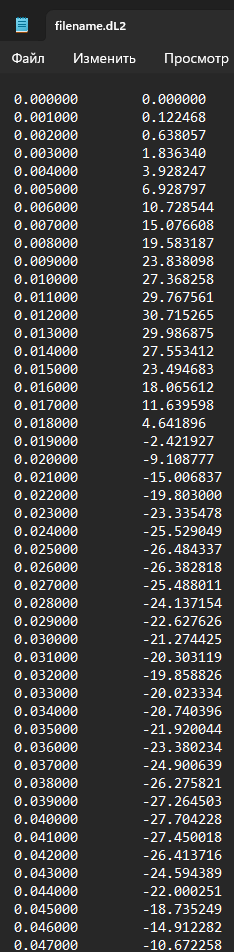
В залежності від обраного типу сигналу виконуються відповідні дії, які включають генерацію, фільтрацію та відображення сигналів.

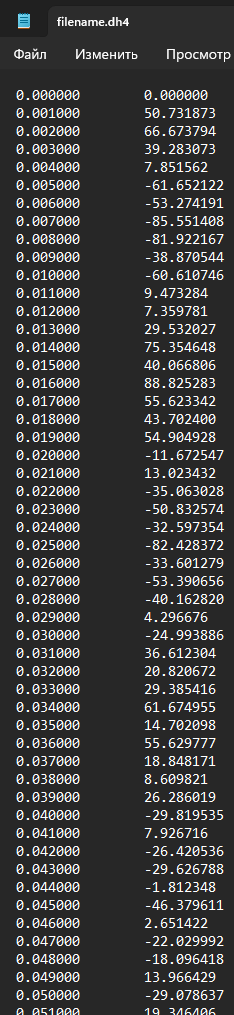
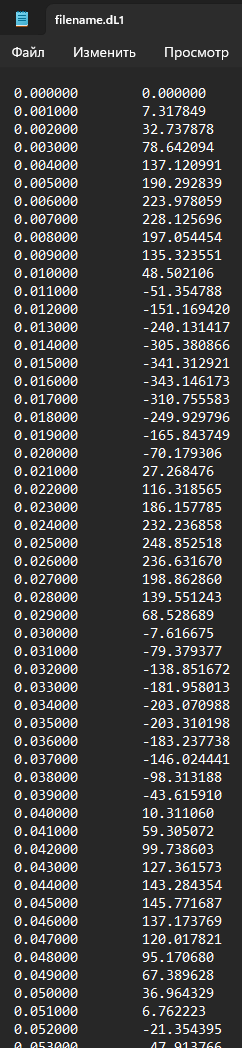
Графіки відображаються з використанням бібліотеки matplotlib.

**Результати роботи**

Створення графіка для тестового сигналу

Створення графіка для робочого сигналу

Збережені даних (повний вміст файла, можна подивитися на [GitHub](https://github.com/FounderC/PAOA-Laboratory))



Посилання на завдання [Replit](https://replit.com/join/inefuuaagh-mrfounder) і на [GitHub](https://github.com/FounderC/PAOA-Laboratory)

**Висновок:** Під час виконання лабораторної роботи я набув практичних навичок розроблення кода для цифрової фільтрації вхідних даних. Було розглянуто створення тестового сигналу на основі заданої математичної моделі, його попереднє розрахування та запис у файл. Також було застосовано низькочастотний та високочастотний фільтри 1-го та 2-го порядку до цього тестового сигналу, і отримані результати були записані у відповідні файли. Для робочого сигналу було здійснено зчитування даних з файлу та їх подальший аналіз.

**Додаток**

MAX\_NUMBER = 65535

BYTE\_ORDER = 'little'  
  
# Створення вхідного тестового сигналу  
def generate\_test\_signal(T, N, dt, frequencies, amplitudes):  
 t = np.linspace(0.0, T, N, endpoint=False)  
 x = np.zeros(N)  
 for f, A in zip(frequencies, amplitudes):  
 x += A \* np.sin(2 \* np.pi \* f \* t)  
 return t, x  
  
# Фільтрація сигналу  
def filter\_signal(signal, lowcut, highcut, fs, order=2):  
 nyquist = 0.5 \* fs  
 low = lowcut / nyquist  
 high = highcut / nyquist  
 b, a = butter(order, [low, high], btype='band')  
 y = lfilter(b, a, signal)  
 return y  
  
# Зчитування даних з 001.dat файлу  
def parse\_sequence(filepath: Union[Path, str]) -> List[int]:  
 chunk\_size = 2  
 y = []  
 with open(filepath, 'rb') as f:  
 while True:  
 chunk = f.read(chunk\_size)  
 if not chunk:  
 break  
 y\_i = int.from\_bytes(chunk, BYTE\_ORDER)  
 if y\_i > 5000:  
 y\_i = y\_i - MAX\_NUMBER  
 y.append(y\_i)  
 return y  
  
signal\_type = input("Виберіть тип сигналу (тестовий/робочий): ")  
  
if signal\_type.lower() == "тестовий":  
 frequencies = [50, 60, 400]  
 amplitudes = [220, 110, 36]  
 T = 1.0  
 N = 1000  
 dt = 1.0 / N  
  
 t, x = generate\_test\_signal(T, N, dt, frequencies, amplitudes)  
  
 lowcut = 10  
 highcut = 100  
  
 # Зберігання результатів у файлах  
 np.savetxt('C:/Users/Khome/Desktop/ЛабаФайл/filename.tst', np.column\_stack((t, x)), fmt='%0.6f', delimiter='\t')  
  
 filtered\_signal = filter\_signal(x, lowcut, highcut, 1.0 / dt)  
 np.savetxt('C:/Users/Khome/Desktop/ЛабаФайл/filename.dL1', np.column\_stack((t, filtered\_signal)), fmt='%0.6f', delimiter='\t')  
  
 filtered\_signal\_l2 = filter\_signal(x, lowcut, 20, 1.0 / dt) np.savetxt('C:/Users/Khome/Desktop/ЛабаФайл/filename.dL2', np.column\_stack((t, filtered\_signal\_l2)), fmt='%0.6f', delimiter='\t')  
  
 filtered\_signal\_h3 = filter\_signal(x, 20, highcut, 1.0 / dt) np.savetxt('C:/Users/Khome/Desktop/ЛабаФайл/filename.dh3', np.column\_stack((t, filtered\_signal\_h3)), fmt='%0.6f', delimiter='\t')  
  
 filtered\_signal\_h4 = filter\_signal(x, 100, 400, 1.0 / dt)   
 np.savetxt('C:/Users/Khome/Desktop/ЛабаФайл/filename.dh4', np.column\_stack((t, filtered\_signal\_h4)), fmt='%0.6f', delimiter='\t')  
  
 # Відображення графіків  
 t\_min, t\_max = -0.2, 1.2  
 custom\_xticks = [-0.2, -0.15, -0.1, -0.05, 0, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6,  
 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95, 1.0, 1.05, 1.1, 1.15, 1.2]  
 custom\_xlabels = ['-0.2', '-0.15', '-0.1', '-0.05', '0', '0.05', '0.1', '0.15', '0.2', '0.25', '0.3', '0.35', '0.4',  
 '0.45', '0.5', '0.55', '0.6', '0.65', '0.7', '0.75', '0.8', '0.85', '0.9', '0.95', '1.0', '1.05', '1.1', '1.15', '1.2']  
  
 # Найстройка графіка  
 fig = plt.figure()  
 fig.patch.set\_facecolor('#393065')  
 ax = fig.add\_subplot(111)  
 ax.plot(t, x, color='orange', linewidth=2)  
 ax.set\_xlabel('Time S', color='#919193')  
 ax.set\_ylabel('Amplitude V', color='#919193')  
 ax.grid(color='gray', linestyle='--', linewidth=0.5)  
 ax.set\_title('Amplitude - Time', color='white')  
 ax.set\_facecolor('#434376')  
 ax.set\_xticks(custom\_xticks)  
 ax.set\_xticklabels(custom\_xlabels, color='#919193')  
 ax.tick\_params(axis='y', colors='#919193')  
 plt.xlim(t\_min, t\_max)  
 plt.show()  
  
elif signal\_type.lower() == "робочий":  
  
 # Зчитування файла  
 data\_file\_path = 'C:/Users/Khome/Desktop/ЛабаФайл/001.dat'  
  
 # Зчитування сигналу і відображення його в графік  
 y = parse\_sequence(data\_file\_path)  
  
 # Відображення графіка сигналу  
 t = np.linspace(-0.2, 1.2, len(y))  
  
 t\_min, t\_max = -2.5, 3  
 custom\_xticks = [-2.5, -2.25, -2, -1.75, -1.5, -1.25, -1, -0.75, -0.5, -0.25, 0, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2, 2.25, 2.5, 2.75, 3]  
 custom\_xlabels = custom\_xticks  
  
 fig = plt.figure()  
 # Встановлення кольору фону  
 fig.patch.set\_facecolor('#393065')  
  
 # Підграфік  
 ax = fig.add\_subplot(111)  
  
 # Найстройка графіка  
 ax.plot(t, y, color='#2B82D5', linewidth=2)  
 ax.set\_xlabel('Time S', color='#919193')  
 ax.set\_ylabel('Amplitude V', color='#919193')  
 ax.grid(color='gray', linestyle='--', linewidth=0.5)  
 ax.set\_title('Amplitude - Time', color='white')  
 ax.set\_facecolor('#434376')  
 ax.set\_xticks(custom\_xticks)  
 ax.set\_xticklabels(custom\_xlabels, color='#919193')  
 ax.tick\_params(axis='y', colors='#919193')  
 plt.xlim(t\_min, t\_max)  
 ax.axhline(y=0, color='#2B82D5', linestyle='-', linewidth=2, label='Центр')  
 plt.show() # Відображення графіку  
else:  
 print("Ви ввели невірний варіант. Спробуйте ще раз.")