**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

**Федеральное государственное бюджетное образовательное учреждение высшего образования «Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова»**



Факультет **информационных технологий и управления**

Кафедра **«Программное обеспечение вычислительной техники»**

Направление **09.04.01 –** **Информатика и вычислительная техника**

**ОТЧЕТ**

**по Лабораторной работе №1**

по дисциплине: **Программное и аппаратное обеспечение**

**информационных систем**

**Выполнил студент** 1 **курса, группы** ТИСа-о24 Купрысь Н.А.

Фамилия, имя, отчество

**Принял доцент, кандидат технических наук** Рыбалкин А.Д.

Фамилия, имя, отчество

«\_\_\_\_\_» **\_\_**\_\_\_\_\_\_\_\_\_\_\_\_2024 г. **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Подпись

Новочеркасск, 2024 г

**Лабораторная работа №1**

**«Применение CASE-средств для проектирования и разработки программного и аппаратного обеспечения ИС и АС»**

**Цель работы:** Изучить классификацию и применение CASE-средств для разработки программного обеспечения, освоить их практическое использование для проектирования и документирования информационных систем, а также разработать первую часть индивидуального задания.

**Теоретический материал:** CASE-средства (Computer-Aided Software Engineering) — это программные инструменты, которые поддерживают процесс проектирования, разработки и сопровождения программного обеспечения. Они автоматизируют многие этапы жизненного цикла программного продукта, облегчая задачу разработчикам и повышая эффективность работы.

CASE-средства можно разделить на несколько категорий:

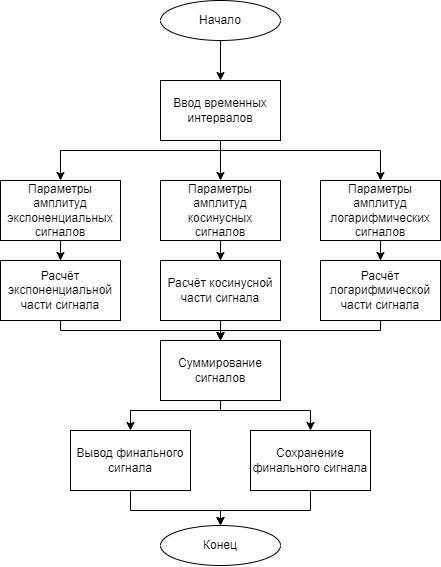
* Средства верхнего уровня — используются для проектирования системы, анализа требований и создания моделей. Примеры: UML-диаграммы, диаграммы потоков данных;
* Средства нижнего уровня — помогают непосредственно в написании кода, тестировании и сопровождении программного обеспечения;
* Интегрированные CASE-средства — объединяют весь процесс разработки, от проектирования до внедрения, поддерживая команды разработчиков на всех этапах.

Кроме того, CASE-средства часто используются для моделирования баз данных, управления проектами и документирования. Ключевой элемент большинства систем — UML, который позволяет визуализировать структуру и поведение программных систем с помощью диаграмм.

Для успешного выбора CASE-инструмента важно учитывать его интеграцию с другими системами, гибкость и возможность масштабирования, а также удобство для команды разработчиков.

**Ход работы:**

1) Разработана диаграмма, отражающая структуру программы. Схема включает три ключевых блока: экспоненциальный компонент, косинусный компонент и логарифмический компонент, которые складываются в общий сигнал, что продемонстрировано на рисунке 1.

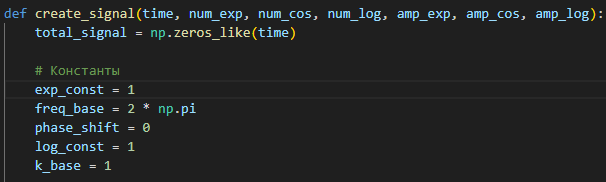


*Рисунок 1 – Структурная схема программы*

2) В первой части кода создается функция create\_signal, которая принимает несколько параметров для генерации сигнала:

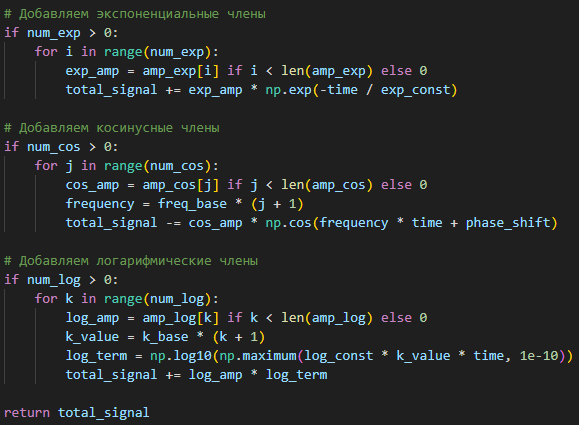
* time: временные точки, для которых будет генерироваться сигнал;
* num\_exp, num\_cos, num\_log: количество экспоненциальных, косинусных и логарифмических членов соответственно;
* amp\_exp, amp\_cos, amp\_log: массивы амплитуд для всех членов.

Инициализируется начальный сигнал с нулевыми значениями, а также задаются константы для экспоненциальной функции, косинусных членов и логарифмических терминов, что показано на рисунке 2.



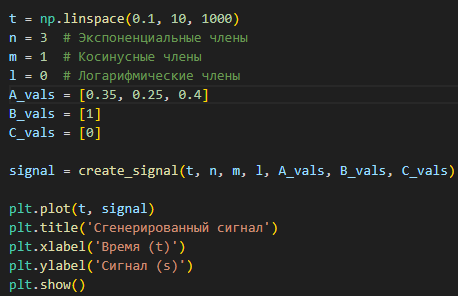
*Рисунок 2 – Функция генерации сигналов и константы*

3) Добавление экспоненциальных, косинусных и логарифмических членов к сигналу, используя заданные формулы. Экспоненциальные члены добавляются для моделирования спадающего сигнала с различными амплитудами, косинусные — для гармонических колебаний, частота которых увеличивается с каждым добавленным членом, а логарифмические — для моделирования сигналов с нелинейным ростом на основе логарифмов. Каждый из этих типов сигналов добавляется поочередно, и итоговый сигнал формируется как сумма всех компонентов, что показано на рисунке 3.



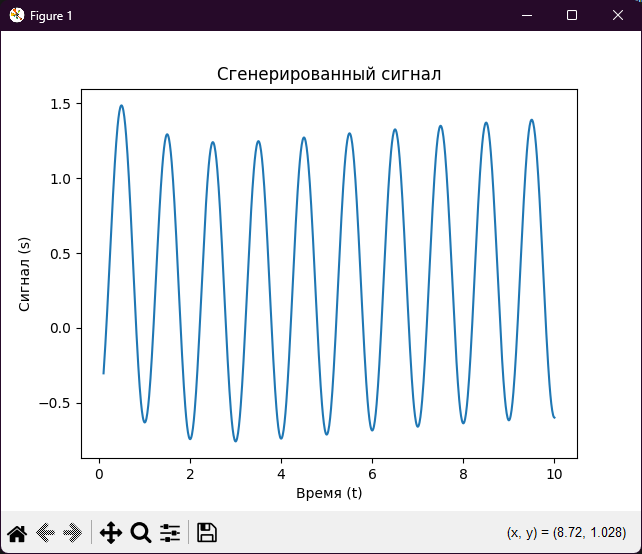
*Рисунок 3 – Добавление компонентов к сигналу*

4) В примере использования функции create\_signal задается временной диапазон от 0.1 до 10 секунд с 1000 равномерно распределенными точками, чтобы избежать проблем с логарифмом от нуля. Затем задаются параметры для каждого типа сигнала: экспоненциальных, косинусных и логарифмических членов. В частности, для экспоненциальных членов указывается, что их два, для косинусных — один, а для логарифмических — один. Соответственно, для каждого типа сигнала заданы массивы амплитуд: для экспоненциальных — два значения, для косинусных и логарифмических — по одному значению. После этого вызывается функция create\_signal, которая использует эти параметры для генерации итогового сложного сигнала, состоящего из всех трёх компонентов, что продемонстрировано на рисунке 4.



*Рисунок 4 – Генерация сложного сигнала*

**Результаты работы:**

После генерации итогового сигнала с помощью функции create\_signal, этот сигнал визуализируется на графике с использованием библиотеки Matplotlib. Вызов функции plt.plot(t, signal) строит график зависимости сигнала от времени, где по оси X откладывается время, а по оси Y — значение сигнала. Заголовок графика задается с помощью plt.title('Сгенерированный сигнал'), а оси подписываются с помощью plt.xlabel('Время (t)') и plt.ylabel('Сигнал').

*Рисунок 5 – Визуализированный сигнал*

**Вывод:** Разработана программа для генерации сложных сигналов, объединяющая экспоненциальные, косинусные и логарифмические компоненты. Программа успешно создает и визуализирует сигнал, позволяя варьировать параметры и амплитуды каждого компонента. Итоговый график подтверждает корректность работы программы и её применимость для моделирования сигналов.