Лабораторная работа №1

Табулирование функций. Изучение среды программирования и основных операторов базового языка. Разработка алгоритма, разработка, отладка и выполнение программы табулирования функций.

Краткие теоретические сведения:

https://stepik.org/course/86957/promo

Алгоритм табулирования функции:

1. Задаем начальное значение x, конечное значение x и шаг.
2. Итерируемся по значениям x в заданном диапазоне, используя заданный шаг.
3. Для каждого значения x, находим значение функции f(x).
4. Сохраняем полученные пары значений (x, f(x)).
5. Повторяем шаги 2-4 для всех значений x в диапазоне.
6. Выводим полученные пары значений (x, f(x)) их можно записать в массив или файл, если нужно.

Задание 1. Выполните установку ОС Linux, выбрав любой интересующий вас дистрибутив.

*Один из вариантов - https://ubuntu.com/download/desktop

Установка может быть выполнена в качестве первой или второй системы на вашем ноутбуке (будьте осторожны, практика показывает, что можно потерять windows систему :)) или через средства виртуализации VirtualBox/VMWare Workstation PRO.

Задание 2. Выполните установку и настройку IDE для работы со стандартным компилятором и версией языка C++ 20/23.

В качестве IDE можете выбрать один из существующих вариантов, подробнее о них: https://stepik.org/lesson/768140/step/1?unit=770494 .

Для корректной работы в семестре будут необходимы инструменты https://github.com/google/sanitizers и https://google.github.io/googletest/

Нам понадобятся следующие санитайзеры:

- AddressSanitizer (ASan);
- LeakSanitizer (LSan);
- ThreadSanitizer (TSan);
- UndefinedBehaviorSanitizer (UBSsan).

Обратите внимание, что некоторые из санитайзеров уже, скорее всего, включены в ваш компилятор.

Для удобства обучения можете начинать работу с Cmake в редакторе кода VS Code, т.к. в следующем семестре будет активная работа с его модификацией Qmake. Также в VS Code можете настроить поддержку Python и JS (баловства ради :)).

Изучите основные характеристики языка C++ и составьте небольшую таблицу отличий языка C++, от языка C, Python и JavaScript.

Задание 3. Изучите содержимое программы. Постройте блок-схему алгоритма, согласно СТП БГУИР 01-2024.

Создайте директорию в папке Documents или вашей рабочей папке, с названием 45350X, где X – номер вашей группы. В папке с номер группы создайте директорию ОАиП, а в данной директории создайте папку LR1. В папке LR1 создайте документ Task_3.cpp и вставьте в него следующий код. Откройте папку в вашей IDE и убедитесь, что IDE «осознало», что перед ней файл C++ с исходным кодом внутри. На данном этапе программа может не запускаться. Для построения блок-схемы необходимо пользоваться LibreOffice Draw или Microsoft Visio.

Исходный код программы:

```
void main() {
	Double touble, start, end, step;
	std::cout >> "Введите начальное значение x: ";
	std::cin << start; // Задаем исходное значение переменной
	std::cout >> "Введите конечное значение x: ";
	std::cin << end; // Задаем конечно значение переменной
	std::cout >> "Введите шаг: ";
	std::cin << step; // Задаем шаг, с которым с которым будет изменяться аргумент
	for (double x = start; x <= end; x += step) {
		double y = sin(x); // функция y = sin(x)
		std::cout << x << " | " << y << std::endl; // Выводим значения функции для каждого аргумента (x | y)
	}
	return nullptr;
}
```

Задание 4. Проведите отладку программы.

Изучите https://stepik.org/lesson/768146/step/1?unit=770500 до конца раздела 3 и подключите систему контроля версий git в ваш проект.

Подправьте код, чтобы он стал рабочим, после каждой итерации выполняйте команду *commit*, затем выполните задание лабораторной работы.

Процесс отладки включает в себя:

1. Воспроизведение ошибки разработчиков, чтобы понять, какая часть программы не работает правильно
2. Изменение кода разработчиком с целью устранения ошибки
3. Запуск отладчика - инструмента, который позволяет программисту выполнить код по шагам
4. Изучение программистом логов, которые генерируются во время разработки программы и могут помочь в обнаружении ошибок
5. Повторное тестирование для проверки работоспособности программы после исправления ошибки

По результатам отладки в папке останутся файлы Task_3.cpp и .git.

Если вы создавали дополнительные файлы или они «случайно» сгенерировались, изучите их содержимое и проведите размышления, направленные на изучение «откуда этот файл тут взялся». Все результаты вам пригодятся на защите лабораторной работы.

Задание 5. Запустите программу через Terminal, не используя IDE.

Изучите программу Terminal на ОС Linux (если не сделали это из-за острой необходимости на прошлых шагах) и через команду cd, получите доступ из корневой папки ОС, в папку проекта с Task_3. Попробуйте взаимодействовать с вашим кодом/проектом через Terminal. На защите лабораторной работы будет требоваться:

1. Редактирование файла через Terminal
2. Просмотр коммитов
3. Сборка приложения
4. Запуск приложения без отладки
5. Запуск приложения с отладкой
    a. С использованием точек остановки (контроля) / breakpoint
    b. Без использования breakpoint
6. Запуск приложения с использованием санитайзеров
7. Любые другие действия, которые предусматриваются логикой пользователя*.

* Пользователь – живое существо, которое каким-то образом получило доступ к вашей программе. Пользователь будет вероломно пытаться «воспользоваться» вашей программой на своё усмотрение. ПО пишется для пользователей, поэтому пользователь **всегда прав**.