**Реквизиты:** Дятлович Анна Анатольевна, 3 курс, Z33434, 2023 год

**Наименование:** ЛР #2: [C++ & UNIX]: C++ BUILD / IF / LOOP, PYTHON

**Цель:** Познакомить студента с принципами компиляции исходного кода. Составить программу с использованием циклов, условий и функций. Сравнить быстродействие между C++ и Python. Ознакомление с типами данных.

**Задача.**

1: [С++ EXPRESSION] Создать и скомпилировать программу на C++

Результат сборки (компиляции) сохранять в папку build. Папку build сделать игнорируемой для GIT. Программа должна получать на вход число – это количество итераций для выполнения расчета. В рамках итерации выполнять следующее вычисление: x ^2- x ^2+ x \*4- x \*5+ x + x . Вычисление выполнять в виде отдельной от main функции, которая будет вызвана циклически из main. Фиксировать время выполнения программы, затрачиваемое на расчет выражения n раз (n задается в консоли перед вычислением). Предусмотреть дополнительный цикл на повторную итерацию запуска программы вычислений. Если было введено не число, то завершить выполнение программы.

**Решение:**

#include <iostream>

#include <chrono>

using namespace std;

double iteracio\_cpp(double x) {

    return x\*x - x\*x + x\*4 - x\*5 + x + x;

}

int main() {

    int n;

    cout << "Print NATURAL NUMBER: ";

    cin >> n;

    if(cin.fail()) {

        cout << "NEVER GONNA GIVE YOU UP" << endl;

        cout << "NEVER GONNA LET YOU DOWN" << endl;

        return 0;

    }

    auto start = chrono::high\_resolution\_clock::now();

    for(int i = 0; i < n; i++) {

        double result = iteracio\_cpp(result);

    }

    auto end = chrono::high\_resolution\_clock::now(); // конец измерения времени выполнения

    auto duration = chrono::duration\_cast<chrono::microseconds>(end - start); // время выполнения в микросекундах

    cout << "Time: " << duration.count() << " microseconds" << endl;

    return 0;

}

В консоли пишем:

g++ lab\_02\_plusi.cpp -o hello

.\hello

**Результат:**

Print NATURAL NUMBER: 1000

Time: 0 microseconds

2 [PYTHON EXPRESSION] Создать и скомпилировать программу на Python 3

Результат сборки (компиляции) сохранять в папку build. Папку build сделать игнорируемой для GIT. Программа должна получать на вход число – это количество итераций для выполнения расчета. В рамках итерации выполнять следующее вычисление: x ^2- x ^2+ x \*4- x \*5+ x + x . Вычисление выполнять в виде отдельной от main функции, которая будет вызвана циклически из main. Фиксировать время выполнения программы, затрачиваемое на расчет выражения n раз (n задается в консоли перед вычислением). Предусмотреть дополнительный цикл на повторную итерацию запуска программы вычислений. Если было введено не число, то завершить выполнение программы.

import time

y = 1

def iteracio(x):

    return x\*\*2 - x\*\*2 + x\*4 - x\*5 + x + x

if \_\_name\_\_ == '\_\_main\_\_':

    try:

        while True:

            n = int(input("Print NATURAL NUMBER:"))

            begin\_time = time.time()\*1000000

            for i in range(int(n)):

                y = iteracio(y)

            end\_time = time.time()\*1000000 - begin\_time

            print('time:', end\_time, 'microsecornds')

    except:

        print('NEVER GONNA GIVE YOU UP\n░░░░░░░░░░░░░░░▄▄░░░░░░░░░░░\n░░░░░░░░░░░░░░█░░█░░░░░░░░░░\n░░░░░░░░░░░░░░█░░█░░░░░░░░░░\n░░░░░░░░░░░░░░█░░█░░░░░░░░░░\n░░░░░░░░░░░░░░█░░█░░░░░░░░░░\n██████▄███▄████░░███▄░░░░░░░')

        print('▓▓▓▓▓▓█░░░█░░░█░░█░░░███░░░░\n▓▓▓▓▓▓█░░░█░░░█░░█░░░█░░█░░░\n▓▓▓▓▓▓█░░░░░░░░░░░░░░█░░█░░░\n▓▓▓▓▓▓█░░░░░░░░░░░░░░░░█░░░░\n▓▓▓▓▓▓█░░░░░░░░░░░░░░██░░░░░')

        print('▓▓▓▓▓▓█████░░░░░░░░░██░░░░░░\n█████▀░░░░▀▀████████░░░░░░░░\nNEVER GONNA LET YOU DOWN')

**Результат:**

Print NATURAL NUMBER:10000

time: 7999.75 microsecornds

Print NATURAL NUMBER:

NEVER GONNA GIVE YOU UP

░░░░░░░░░░░░░░░▄▄░░░░░░░░░░░

░░░░░░░░░░░░░░█░░█░░░░░░░░░░

░░░░░░░░░░░░░░█░░█░░░░░░░░░░

░░░░░░░░░░░░░░█░░█░░░░░░░░░░

██████▄███▄████░░███▄░░░░░░░

▓▓▓▓▓▓█░░░█░░░█░░█░░░███░░░░

▓▓▓▓▓▓█░░░█░░░█░░█░░░█░░█░░░

▓▓▓▓▓▓█░░░░░░░░░░░░░░█░░█░░░

▓▓▓▓▓▓█░░░░░░░░░░░░░░░░█░░░░

▓▓▓▓▓▓█░░░░░░░░░░░░░░██░░░░░

▓▓▓▓▓▓█████░░░░░░░░░██░░░░░░

█████▀░░░░▀▀████████░░░░░░░░

NEVER GONNA LET YOU DOWN

3. [SAVE] Результат всех вышеперечисленных шагов сохранить в репозиторий (+ отчет по данной ЛР в папку doc) Фиксацию ревизий производить строго через ветку dev. С помощью скриптов накатить ревизии на stg и на prd. Скрипты разместить в корне репозитория. Также создать скрипты по возврату к виду текущей ревизии (даже если в папке имеются несохраненные изменения + новые файлы).

**Заключение:** C++ быстрый и интересный язык. С помощью try/except в Python3 можно делать весёлые ошибки.