**Лабораторная работа №15**

Выбор и обоснование выбора среды разработки программы. Изучение различных стилей программирования, правил формирования листинга программы.

**Задание 1**

Для разработки ПП я буду использовать ЯП C#, в частности его ответвление для Unity.

C# – это объектно-ориентированный, простой и в то же время мощный язык программирования, который позволяет разработчикам создавать многофункциональные приложения.

**Задание 2**

Для моего проекта лучше всего подойдет объектно-ориентирование программирование, так как выбранный ЯП и среда разработки ориентированы на данную парадигму программирования.

**Задание 3**

**Структурное программирование:**

**Условие:** Напишите функцию collatz (), принимающую один параметр: number. Если number —четное число, функция collatz () должна вывести на экран и возвратить значение number // 2. Если же number— нечетное число, то функция должна вывести на экран и возвратить значение 3 \* number + 1.

После этого напишите программу, которая предлагает пользователю ввести целое число, а затем последовательно вызывает функцию collatz () для этого числа и значений, возвращаемых очередным вызовом этой функции, пока на каком-то этапе не будет возвращено значение 1. (Любопытно отметить, что, независимо от выбора начального числа, вы все равно рано или поздно получите l! Даже математики не могут объяснить, почему так происходит. Числовая последовательность,

которую вы исследуете с помощью этой программы, называется последовательностью Коллатца и иногда характеризуется как &quot;простейшая из неразрешенных проблем математики&quot;.)

Предусмотреть проверку корректности ввода с помощью инструкции try и except. Не забывайте о том, что возвращаемое функцией input () значение нуждается в преобразовании в целое число с помощью функции int (), иначе это будет строковое значение.

def cotllaz(number): # функция

if number%2 == 0:

return(number//2)

if number%2 == 1:

return(3\*number+1)

a = 0 # основной код

try: # проверка на тип вводимого значения

a = int(input('Введите число: ')) # ввод числа

while a != 1: #цикл

a = cotllaz(a) # вызов функции

print(a)

except ValueError:

print('Введено неверное значение!') # вывод при неверном значении

**Модульное программирование:**

**Условие:** Написать программу, с использованием нескольких функций.

def hello():# первая функция

print('Hello, world!')

def fib(n): #вторая функция

a = b = 1

for i in range(n - 2):

a, b = b, a + b

return b

if \_\_name\_\_ == "\_\_main\_\_":

hello()# вызов первой функции

for i in range(10):

print(fib(i)) #вызов второй функции в цикле

**Объектно-ориентированное программирование:**

**Условие:** Создать класс Specialist, конструктор которого принимает три параметра (не

учитывая self) – имя, фамилию и квалификацию специалиста. Квалификация имеет

значение заданное по умолчанию, равное единице. У класса есть метод, который возвращает строку, включающую в себя всю информацию о специалисте. Класс содержит деструктор, который выводит на экран фразу «До свидания, мистер …» (вместо троеточия должны выводиться имя и фамилия объекта). В основной ветке программы создайте три объекта класса Specialist. Посмотрите

информацию о сотрудниках и увольте «самое слабое звено».

class Specialist: # название класса

def \_\_init\_\_(self,fname,lname,level = 1): # конструктор класса

self.fname = fname

self.lname = lname

self.level = level

def print\_info(self): # функция вывода информации

print('Имя:', self.fname, ' Фамилия:', self.lname, ' Квалификация: ', self.level)

def \_\_del\_\_(self): # деструктор класса

print('До свидания мистер', self.lname)

a = Specialist('Дмитрий','Кормильчик',3) # создание экземпляров класса

b = Specialist('Максим','Мулярчик',2)

c = Specialist('Александр','Муханюк')

a.print\_info()# использование функций класса

b.print\_info()

c.print\_info()

input()

del c

input()