Студент группы ИС-23, Сидин Николай.

Практическое занятие №17.

Тема: составление программ с использованием ООП.

Цель: закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составления программ с использованием ООП в IDE PyCharm Community.

Постановка задач:
*1. Создайте класс "Счётчик", который имеет атрибут текущего значения и методы для инкремента и декремента значения.*

*2. Создание базового класса "Работник" и его наследование для создания классов "Менеджер" и "Инженер". В классе "Работник" будут общие методы, такие как "работать" и "получать зарплату", а классы-наследники будут иметь свои уникальные методы и свойства, такие как "управлять командой" и "проектировать системы".*

Текст программы:
1.

```
# Вариант 20. Создайте класс "Счётчик", который имеет
# атрибут текущего значения и методы для инкремента и декремента значения.

class Counter:
    def __init__(self, number):
        self.number = number

    def increment_number(self, new_number):
        self.number += new_number
        return f"Значение после прибавления new_number: {self.number}"

    def dicrement_number(self, new_number):
        self.number -= new_number
        return f"Значение после вычитания new_number: {self.number}"


obj = Counter(30)
obj_2 = Counter(10)
print(obj.increment_number(3))
print(obj.dicrement_number(10))
print()
print(obj_2.increment_number(4))
print(obj_2.dicrement_number(10))
```

2.

```
# Вариант 20. Создание базового класса "Работник" и его наследование
# для создания классов "Менеджер" и "Инженер". В классе "Работник" будут
# общие методы, такие как "работать" и "получать зарплату", а классы-наследники
# будут иметь свои уникальные методы и свойства, такие как
# "управлять командой" и "проектировать системы".


class WorkMan:
    def work(self, do: True or False):
        self.worked = do
        return f"Работают: {self.worked}"

    def get_zarplat(self, zr: True or False):
        self.take_zarplat = zr
        return f"Получают зарплату: {self.take_zarplat}"


class Manager(WorkMan):
    def __init__(self, zarplata):
        self.zarplata = zarplata
        self.otdel = "Менеджер"

    def manage_command(self, manage: True or False):
        self.manage = manage
        return f"Управление командой: {self.manage}"

    def project_system(self, ans: True or False):
        self.yprav = ans
        return f"Проектирование системы: {self.yprav}"

class Injener(WorkMan):
    def __init__(self, zarplata):
        self.zarplata = zarplata
        self.otdel = 'Инженер'


    def manage_command(self, manage: True or False):
        self.manage = manage
        return f"Управление командой: {self.manage}"

    def project_system(self, ans: True or False):
        self.yprav = ans
        return f"Проектирование системы: {self.yprav}"

Man_1 = Manager(19000)
Man_2 = Injener(40000)
print(Man_1.manage_command(True))
print(Man_1.project_system(False))
print(Man_1.get_zarplat(True))
print(Man_1.work(True))
```

```
print()

print(Man_2.manage_command(False))
print(Man_2.project_system(False))
print(Man_2.get_zarplat(True))
print(Man_2.work(True))
```

Протокол работы программы:

```
1. Значение после прибавления new_number: 33
Значение после вычитания new_number: 23

Значение после прибавления new_number: 14
Значение после вычитания new_number: 4

Process finished with exit code 0

2. Управление командой: True
Проектирование системы: False
Получают зарплату: True
Работают: True

Управление командой: False
Проектирование системы: False
Получают зарплату: True
Работают: True

Process finished with exit code 0
```

Вывод: в процессе выполнения практического задания я закрепил усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрел навыки составления программ с использованием ООП в IDE PyCharm Community.