

## Практическое занятие №17

**Тема:** составление программ с использованием ООП.

**Цели практического занятия:**

Закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составления программ с ООП в IDE PyCharm Community.

**Постановка задачи 1:**

Создайте класс «Матрица», который имеет атрибуты количества строк и столбцов. Добавьте методы для сложения, вычитания и умножения матриц.

**Текст программы 1:**

```
# Создайте класс «Матрица», который имеет атрибуты
количества строк и столбцов.
# Добавьте методы для сложения, вычитания и умножения
матриц.

from random import randint

class Matrix:

    def __init__(self, rows, cols):
        self.rows = rows
        self.cols = cols
        self.grid = [[]]

    def __str__(self):
        return '\n'.join(['\t'.join([str(j) for j in i])
for i in self.grid]) + '\n'

    @staticmethod
    def out(lst):
        return '\n'.join(['\t'.join([str(j) for j in i])
for i in lst]) + '\n'
```

```
def __add__(self, other):  
    return self.out([[self.grid[i][j] +  
other.grid[i][j] for j in range(self.cols)] for i in  
range(self.rows)])
```

```
def __sub__(self, other):  
    return self.out([[self.grid[i][j] -  
other.grid[i][j] for j in range(self.cols)] for i in  
range(self.rows)])
```

```
def __mul__(self, other):  
    return self.out([[sum([self.grid[i][k] *  
other.grid[k][j] for k in range(self.cols)]) for j in  
range(self.cols)]  
                    for i in range(self.rows)])
```

```
# Создание матриц  
a = Matrix(3, 3)  
a.grid = [[randint(1, 10) for _ in range(a.cols)] for _  
in range(a.rows)]  
print('Матрица a:')  
print(a)
```

```
b = Matrix(3, 3)  
b.grid = [[randint(1, 10) for _ in range(b.cols)] for _  
in range(b.rows)]  
print('Матрица b:')  
print(b)
```

```
print('Сложение матриц:')  
print(a + b)
```

```
print('Вычитание матриц:')  
print(a - b)
```

```
print('Умножение матриц:')  
print(a * b)
```

## Протокол работы программы 1:

```
PZ_17_1 x
"C:\Program Files\Python310\python.exe
Матрица a:
3  4  3
5  2  3
10 1  9

Матрица b:
5  1  1
4  3 10
8  8 10

Сложение матриц:
8  5  4
9  5 13
18 9 19

Вычитание матриц:
-2  3  2
1  -1 -7
2  -7 -1

Умножение матриц:
55 39 73
57 35 55
126 85 110

Process finished with exit code 0
```

## Постановка задачи 2:

Создание базового класса "Транспортное средство" и его наследование для создания классов "Автомобиль" и "Мотоцикл". В классе "Транспортное средство" будут общие свойства, такие как максимальная скорость и количество колес, а классы наследники будут иметь свои уникальные свойства и методы.

## Текст программы 2:

```
# Создание базового класса "Транспортное средство" и его
наследование для создания классов "Автомобиль" и
"Мотоцикл".
# В классе "Транспортное средство" будут общие свойства,
такие как максимальная скорость и количество колес,
# а классы наследники будут иметь свои уникальные
свойства и методы.

class Transport:
    def __init__(self, max_speed, num_wheels):
        self.max_speed = max_speed
        self.num_wheels = num_wheels

    def init(self, max_speed, num_wheels):
        pass

class Car(Transport):
    def __init__(self, max_speed, num_wheels, brand,
model):
        super().__init__(max_speed, num_wheels)
        super().init(max_speed, num_wheels)
        self.brand = brand
        self.model = model

    def start_engine(self):
        print(
            f"Это {self.brand} {self.model} двигатель
работает, её максимальная скорость {self.max_speed},
количество "
```

```
f"колес {self.num_wheels}")
```

```
class Motorcycle(Transport):  
    def __init__(self, max_speed, num_wheels, brand,  
model):  
        super().__init__(max_speed, num_wheels)  
        super().__init__(max_speed, num_wheels)  
        self.brand = brand  
        self.model = model
```

```
    def wheelie(self):  
        print(f"Это {self.brand} {self.model}, его  
скорость {self.max_speed}, количество колес  
{self.num_wheels}")
```

```
# Создаем объекты классов Car и Motorcycle  
my_car = Car(200, 4, "Toyota", "Corolla")  
my_motorcycle = Motorcycle(150, 2, "Harley-Davidson",  
"Sportster")
```

```
# Вызываем методы start_engine и wheelie  
my_car.start_engine()  
my_motorcycle.wheelie()
```

## Протокол работы программы 2:

```
PZ_17_2 x  
"C:\Program Files\Python310\python.exe" C:\Users\honor\Desktop\Программирование\Proj_1sem_Gorozhy\PZ_17\PZ_17_2.py  
Это Toyota Corolla двигатель работает, её максимальная скорость 200, количество колес 4  
Это Harley-Davidson Sportster, его скорость 150, количество колес 2  
  
Process finished with exit code 0
```

## Вывод:

В ходе выполнения данного практического занятия закрепила усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрела навыки составления программ с ООП в IDE PyCharm Community.

Выполнены разработка кода, отладка, тестирование, оптимизация программного кода.

Готовые программные коды выложены на GitHub.