

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

ФАКУЛЬТЕТ ЭЛЕКТРОННО-ИНФОРМАЦИОННЫХ СИСТЕМ

Кафедра интеллектуальных информационных технологий

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

По дисциплине “Системное программирование”

Специальность “Программная инженерия”

Выполнил:

Е.И. Мицкевич

Студент группы ПО-13

Проверил:

А.Д. Кулик

преп.-стажёр кафедры ИИТ

Цель работы: закрепить навыки объектно-ориентированного программирования на языке Python

Ход работы

4) Прямоугольник, заданный длинами двух сторон – Предусмотреть возможность определения площади и периметра, а также логические методы, определяющие, является ли прямоугольник квадратом и существует ли такой прямоугольник. Конструктор должен позволять создавать объекты с начальной инициализацией. Переопределить метод `__eq__`, выполняющий сравнение объектов данного типа.

```
From enum import Enum

class Rectangle:
    def __init__(self, sideA, sideB):
        if (sideB>0 or sideA>0):
            self._sideA=sideA
            self._sideB=sideB
        else:
            print("Неверный ввод данных, используются значения по умолчанию")
            self._sideA=5
            self._sideB=10

    def square(self):
        return self._sideA*self._sideB

    def perimeter(self):
        return 2*(self._sideA+self._sideB)

    def isKvadrat(self):
        if self._sideA==self._sideB:
            return True

        return False

    def isReal(self):
        if (self._sideB<=0 or self._sideA<=0):
            return False
        return True

    def __eq__(self, other):
        if (self.square() == other.square()):
            return True
        return False

class Menu(Enum):
    CREATE=1
    SQUARE=2
    PERIM=3
    IS_KVAD=4
    IS_REAL=5
    EQ=6
    EXT=7

print("МЕНЮ")
print(f"{Menu.CREATE.value}) Создать прямоугольник")
```

```

print(f"{Menu.SQUARE.value}) Посчитать площадь")
print(f"{Menu.PERIM.value}) Посчитать периметр")
print(f"{Menu.IS_KVAD.value}) Квадрат ли это")
print(f"{Menu.IS_REAL.value}) Может ли он существовать")
print(f"{Menu.EQ.value}) Сравнить площади")
print(f"{Menu.EXT.value}) Выход")

rect=None
rect_2=None

while True:
    chouse=int(input("\nСделайте выбор "))
    match chouse:
        case Menu.CREATE.value:
            sideA=int(input("Введите сторону A"))
            sideB=int(input("Введите сторону B"))
            rect=Rectangle(sideA, sideB)

        case Menu.SQUARE.value:
            if rect:
                print(f"Площадь = {rect.square()}")
            else:
                print("Сначала создайте прямоугольник!")

        case Menu.PERIM.value:
            if rect:
                print(f"Периметр = {rect.perimeter()}")
            else:
                print("Сначала создайте прямоугольник!")

        case Menu.IS_KVAD.value:
            if rect:
                print(f"Квадрат ли это? {rect.isKvadrat()}")
            else:
                print("Сначала создайте прямоугольник!")

        case Menu.IS_REAL.value:
            if rect:
                print(f"Может ли существовать? {rect.isReal()}")
            else:
                print("Сначала создайте прямоугольник!")

        case Menu.EQ.value:
            if rect:
                sideA=int(input("Введите сторону A для второго прямоугольника: "))
                sideB=int(input("Введите сторону B для второго прямоугольника: "))
                rect_2=Rectangle(sideA, sideB)
                print(f"Равны ли площади? {rect==rect_2}")
            else:
                print("Сначала создайте первый прямоугольник!")

        case Menu.EXT.value:
            print("ДО СВИДАНИЯ!")
            break

        case _:
            print("Неверный выбор! Попробуйте снова.")

```

```

МЕНЮ
1) Создать прямоугольник
2) Посчитать площадь
3) Посчитать периметр
4) Квадрат ли это
5) Может ли он существовать
6) Сравнить площади
7) Выход

Сделайте выбор 1
Введите сторону А3
Введите сторону В5

Сделайте выбор 2
Площадь = 15

Сделайте выбор 3
Периметр = 16

Сделайте выбор 4
Квадрат ли это? False

Сделайте выбор 5
Может ли существовать? True

Сделайте выбор 6
Введите сторону А для второго прямоугольника: 4
Введите сторону В для второго прямоугольника: 6
Равны ли площади? False

Сделайте выбор 7
ДО СВИДАНИЯ!
PS C:\Users\Huawei>

```

3) Система Больница. Пациенту назначается лечащий Врач. Врач может сделать назначение Пациенту (процедуры, лекарства, операции). Медсестра или другой Врач выполняют назначение. Пациент может быть выписан из Больницы по окончании лечения, при нарушении режима или иных обстоятельствах.

```

class Hospital:
    def __init__(self, title):
        self.title=title
        self.doctors=[]
        self.patients=[]

    def manage(self):
        print(f"--- Работа больницы '{self.title}' ---")

class MedicalStaff:
    def __init__(self, name):
        self.name=name

class Doctor(MedicalStaff):

    def __init__(self, name, specializacia):
        super().__init__(name)
        self.specializacia=specializacia

    def creatNaznachenie(self, pacient, typeNazn, nameNazn):
        print(f"Врач {self.name} назначил {typeNazn}: {nameNazn} для {pacient.name}")
        pacient.addPrescription(typeNazn, nameNazn)

    def doOperation(self, pacient):
        if "operation" in pacient.prescriptions and pacient.prescriptions
["operation"]:
##первое - есть ли ключ оператион в мапе, второе - есть ли там значение
        opName=pacient.prescriptions["operation"].pop(0)
        print(f"Врач {self.name} проводит операцию: {opName} пациенту
{pacient.name}")
        else:

```

```

        print(f"У пациента {pacient.name} нет назначенных операций.")

class Pacient:
    def __init__(self, name):
        self.name = name
        self.prescriptions = {
            "procedur": [],
            "lekcarstvo": [],
            "operation": []
        }
        self.isDischarged = False
        self.lechaushiyVarch = None

    def addPrescription(self, category, name):
        if category in self.prescriptions:
            self.prescriptions[category].append(name)

    def setDoctor(self, doctor):
        self.lechaushiyVarch = doctor
        print(f"Пациенту {self.name} назначен лечащий врач: {doctor.name}")

    def discharge(self, reason):
        self.isDischarged = True
        print(f"Пациент {self.name} выписан. Причина: {reason}")

class Medsister(MedicalStaff):
    def giveLeckarstvo(self, pacient):
        if "lekcarstvo" in pacient.prescriptions and pacient.prescriptions["lekcarstvo"]:
            med = pacient.prescriptions["lekcarstvo"].pop(0)
            print(f"Медсестра {self.name} выдала лекарство {med} пациенту {pacient.name}")
        else:
            print(f"Для пациента {pacient.name} нет лекарств.")

    def doProcedur(self, pacient):
        if "procedur" in pacient.prescriptions and pacient.prescriptions["procedur"]:
            proc = pacient.prescriptions["procedur"].pop(0)
            print(f"Медсестра {self.name} выполнила процедуру {proc} пациенту {pacient.name}")
        else:
            print(f"Для пациента {pacient.name} нет процедур.")

# Создаем персонал
doctor1 = Doctor("Фолитарик", "Хирург")
medSister1 = Medsister("Яна")

# Создаем пациента
pacient1 = Pacient("Шрамук")

# Назначаем лечащего врача (Ассоциация)
pacient1.setDoctor(doctor1)

# Врач делает назначения
doctor1.creatNaznachenie(pacient1, "lekcarstvo", "Аспирин")
doctor1.creatNaznachenie(pacient1, "procedur", "Прогревание")
doctor1.creatNaznachenie(pacient1, "operation", "Удаление аппендицита")

print("""*30)

# Медсестра выполняет назначения
medSister1.giveLeckarstvo(pacient1)
medSister1.doProcedur(pacient1)

# Врач выполняет операцию

```

```
doctor1.doOperation(patient1)

# Выписка пациента
print("""30)
patient1.discharge("Завершение лечения")
```

```
Пациенту Шрамук назначен лечащий врач: Фолитарик
Врач Фолитарик назначил lekarstvo: Аспирин для Шрамук
Врач Фолитарик назначил procedur: Прогревание для Шрамук
Врач Фолитарик назначил operation: Удаление аппендицита для Шрамук
*****
Медсестра Яна выдала лекарство Аспирин пациенту Шрамук
Медсестра Яна выполнила процедуру Прогревание пациенту Шрамук
Врач Фолитарик проводит операцию: Удаление аппендицита пациенту Шрамук
*****
Пациент Шрамук выписан. Причина: Завершение лечения
```

Вывод: были отработаны базовые знания ООП языка программирования Python при решении практических задач.