

«Министерство образования и науки Российской
Федерации Федеральное государственное бюджетное
образовательное
учреждение высшего образования «Алтайский государственный
технический университет им. И.И. Ползунова»

Факультет информационных технологий
Кафедра прикладной математики

Отчёт защищён с оценкой _____
Преподаватель Савченко В.В.

«_____» _____ 2022 г.

Отчёт
Лабораторной работе №5
«Структурные паттерны 1 - проектирование
(Adapter, Decorator)»

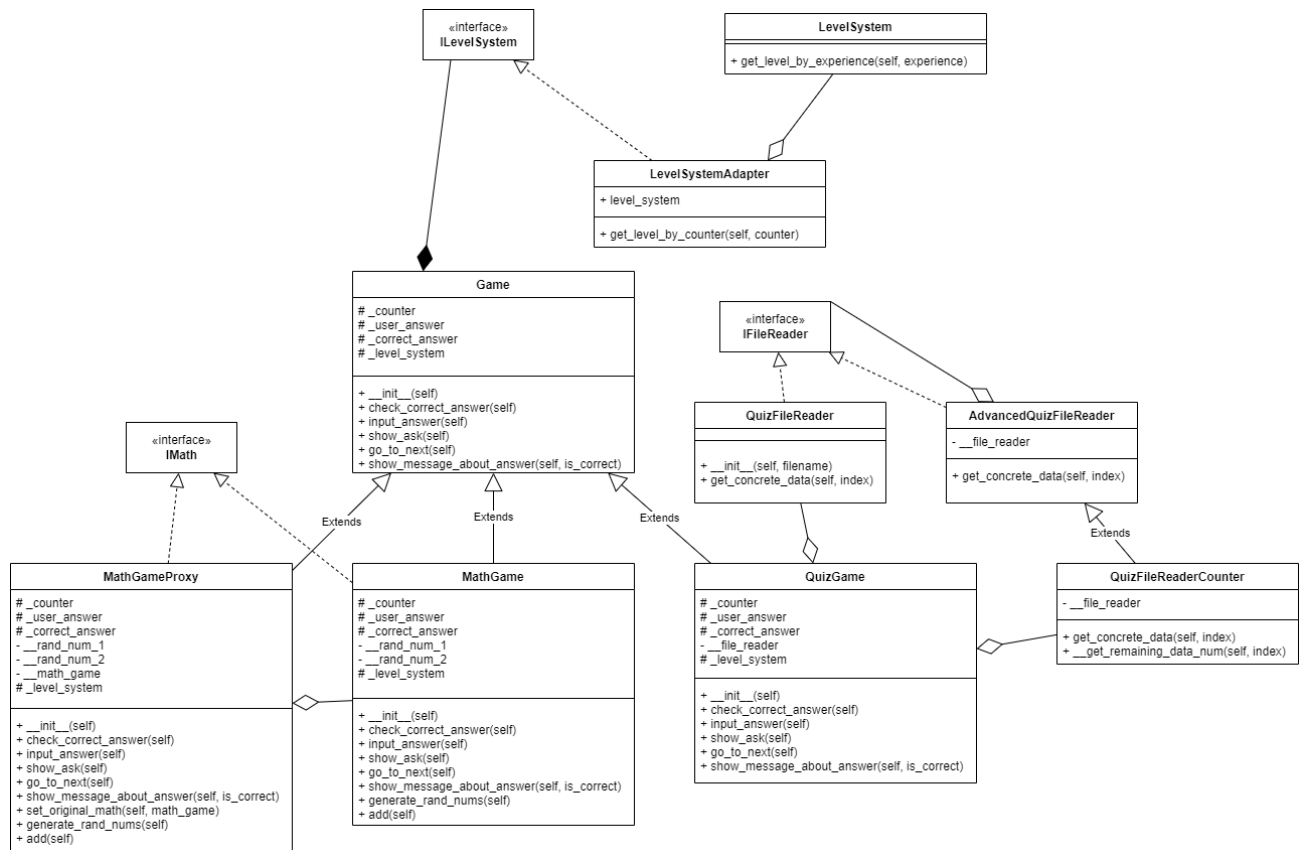
Студент группы ПИ 92 В.М. Шульпов

Преподаватель Савченко В.В.

Задание

№	ФИО студента группы ПИ-92	Прикладная область	Задание
219	Шульпов Виктор Максимович	Игровые системы	Проектирование обучающего игрового комплекса для младших школьников

Диаграмма классов

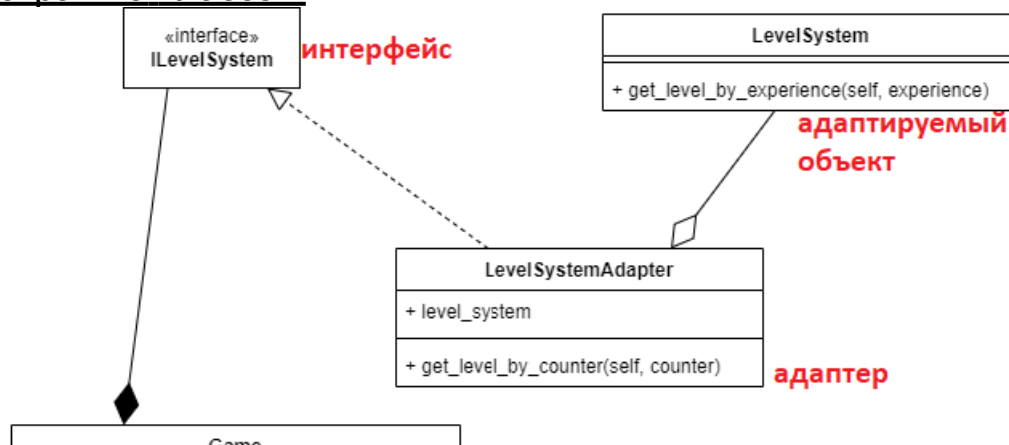


Краткое описание архитектуры паттерна АДАПТЕР в программе

Проблема: необходимо обеспечить взаимодействие несовместимых компонентов с разными интерфейсами. У нас есть класс `LevelSystem`, в котором реализуется получение значения текущего уровня с параметром `experience` (опыт). Однако клиентский интерфейс ничего не знает про `experience`, но знает про `counter` (счетчик выполненных заданий).

Решение: конвертировать исходный интерфейс компонента к другому виду с помощью промежуточного объекта - адаптера. Класс Адаптер реализует интерфейс, известный его клиентам.

Диаграмма классов:



- Адаптер (`LevelSystemAdapter`) реализует интерфейс `ILevelSystem`, на который рассчитывает клиент
- Адаптер (`LevelSystemAdapter`) содержит ссылку на

- адаптируемый объект (LevelSystem)
- Адаптер реализует все методы

интерфейса Код интерфейса (ILevelSystem):

```
from abc import ABC, abstractmethod

class ILevelSystem(ABC):
    """интерфейс уровневой системы"""

    @abstractmethod
    def get_level_by_counter(self, counter):
        """посчитать текущий уровень по количеству пройденных уровней"""
        pass
```

Код адаптера (LevelSystemAdapter):

```

from ilevel_system import ILevelSystem

class LevelSystemAdapter(ILevelSystem):
    """адаптер уровневой системы"""
    def __init__(self, level_system):
        self.level_system = level_system

    def get_level_by_counter(self, counter):
        """посчитать текущий уровень по количеству пройденных уровней
        1 пройденное задание = 50 опыта
        100 опыта = 1 уровень"""
        experience = 50 * counter
        return self.level_system.get_level_by_experience(experience)

```

Код адаптируемого объекта (LevelSystem):

```

class LevelSystem():
    """уровневая система"""
    def __init__(self):
        pass

    def get_level_by_experience(self, experience):
        """посчитать текущий уровень по количеству опыта
        100 опыта = 1 уровень"""
        return experience // 100

```

Краткое описание архитектуры паттерна ДЕКОРАТОР в программе

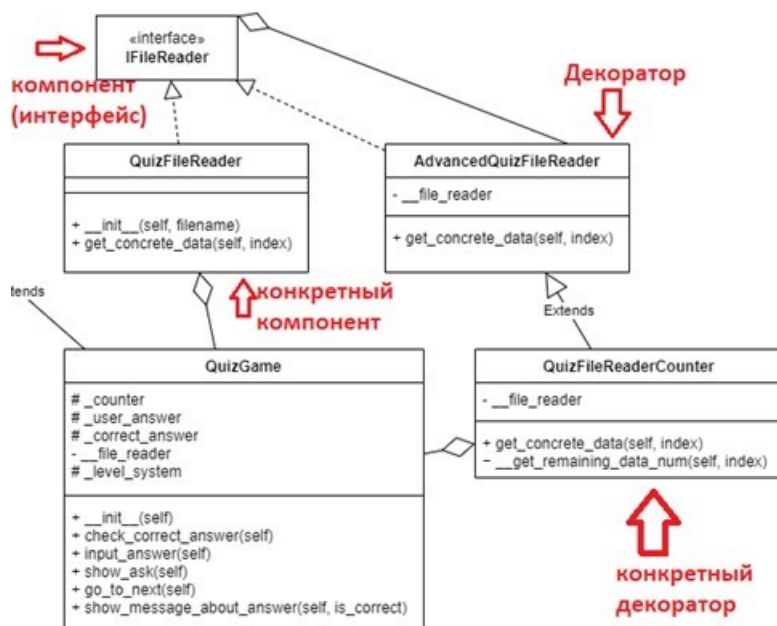
Проблема: Необходимо возложить дополнительные обязанности на отдельный объект, а не на класс в целом. Надо имитировать интерфейс у объекта, который таким интерфейсом не обладает.

Решение: динамически добавить объекту новые обязанности, не прибегая при этом к порождению подклассов.

Результат: большая гибкость, чем у статического наследования: можно добавлять и удалять обязанности во время выполнения программы, в то время как при использовании наследования надо было бы создавать новый класс для каждой дополнительной обязанности.

Данный паттерн позволяет избежать перегруженных методами классов на верхних уровнях иерархии — новые обязанности можно добавлять по мере необходимости

Диаграмма классов:



- Декоратор (AdvancedQuizFileReader) реализует тот же интерфейс, что и конкретный компонент (QuizFileReader).
- Декоратор (AdvancedQuizFileReader) содержит ссылку на декорируемый объект (IFileReader)

- Декоратор (AdvancedQuizFileReader) дополняет функционал декорируемого объекта (IFileReader)

Код интерфейса (IFileReader):

```
from abc import ABC, abstractmethod
```

```
class IFileReader(ABC):
    """читатель файла"""
    @abstractmethod
    def get_concrete_data(self, index):
        """получение данных для уровня"""
    pass
```

Код конкретного компонента (QuizFileReader)

```
from ifile_reader import IFileReader
import csv
```

```
class QuizFileReader(IFileReader):
    """читатель файла для игры викторина (читает вопрос, варианты ответа, и номер правильного ответа)"""
    def __init__(self, filename):
        self._filename = filename

    def get_filename(self):
        """getter filename"""
        return self._filename

    def get_concrete_data(self, index):
        """получать вопрос, номер правильного ответа и варианты ответа"""

        with open(self._filename, 'r', newline='') as File:
            reader = csv.reader(File, delimiter=',')
            counter = 0
            for ask_i, row in enumerate(reader):
                ask = row[0]
                correct_answer_num = next(reader)[0]
                answers = next(reader)
                counter += 1
                if ask_i == index:
                    return [ask, correct_answer_num, answers]
            return None
```

Код декоратора (AdvancedQuizFileReader):

```
from ifile_reader import IFileReader
import csv
```

```
class AdvancedQuizFileReader(IFileReader):
    """читатель-декоратор файла для игры викторина
    (читает вопрос, варианты ответа, и номер правильного ответа)
    дополняет функционал за счет способности подсчитывать количество оставшихся в файле уровней"""
    def __init__(self, file_reader):
        pass
```

```

def get_concrete_data(self, index):
    """получать вопрос, номер правильного ответа и варианты ответа"""
    pass

def get_filename(self):
    """getter filename"""
    pass

def __get_remaining_data_num(self, index):
    """подсчитывает количество оставшихся вопросов в файле"""
    pass

```

Код конкретного декоратора (QuizFileReaderCounter):

```

from advanced_quiz_file_reader import AdvancedQuizFileReader
import csv

class QuizFileReaderCounter(AdvancedQuizFileReader):
    """читатель-декоратор файла для игры викторина
    (читает вопрос, варианты ответа, и номер правильного ответа)
    дополняет функционал за счет способности подсчитывать количество оставшихся в файле уровней"""
    def __init__(self, file_reader):
        self._file_reader = file_reader

    def get_concrete_data(self, index):
        """получать вопрос, номер правильного ответа и варианты ответа"""
        counter = self._get_remaining_data_num(index)
        print(f'Осталось вопросов: {counter}')
        return self._file_reader.get_concrete_data(index)

    def get_filename(self):
        """getter filename"""
        return self._file_reader.get_filename()

    def __get_remaining_data_num(self, index):
        """подсчитывает количество оставшихся вопросов в файле"""
        with open(self._file_reader.get_filename(), 'r', newline='') as File:
            reader = csv.reader(File, delimiter=',')
            counter = 0
            for ask_i, row in enumerate(reader):
                next(reader)
                next(reader)
                if ask_i < index:
                    continue
                counter += 1
            return counter

```


Код программы:

game.py

```
from level_system_adapter import LevelSystemAdapter
from level_system import LevelSystem

class Game:
    """игра"""
    def __init__(self):
        self._counter = 0

        self._user_answer = None
        self._correct_answer = None
        self._level_system = LevelSystemAdapter(LevelSystem())

    def check_correct_answer(self):
        """проверка правильности ответа пользователя"""
        if self._user_answer == self._correct_answer:
            return True
        return False

    def input_answer(self):
        """ввод пользовательского ответа"""
        pass

    def show_ask(self):
        """показ вопроса"""
        pass

    def go_to_next(self):
        """увеличение счетчика (переход на следующий уровень/вопрос)"""
        self._counter += 1
        print(f'Переход на следующее задание!\nВаш текущий уровень {self.get_level()}!')

    def show_message_about_answer(self, is_correct):
        """сообщение о правильности ответа"""
        if is_correct:
            print(f'Вы ответили правильно! (ответ: {self._correct_answer})')
        else:
            print('Вы ответили неверно!')

    def get_level(self):
        """получить текущий уровень"""
        return self._level_system.get_level(self._counter)
```

ifile_reader.py:

```
from abc import ABC, abstractmethod
```

```
class IFileReader(ABC):  
    """читатель файла"""  
    @abstractmethod  
    def get_concrete_data(self, index):  
        """получение данных для уровня"""  
        pass
```

ilevel_system.py:

```
from abc import ABC, abstractmethod
```

```
class ILevelSystem(ABC):  
    """интерфейс уровневой системы"""  
  
    @abstractmethod  
    def get_level_by_counter(self, counter):  
        """посчитать текущий уровень по количеству пройденных уровней"""  
        pass
```

level_system.py:

```
class LevelSystem():  
    """уровневая система"""  
    def __init__(self):  
        pass  
  
    def get_level_by_experience(self, experience):  
        """посчитать текущий уровень по количеству опыта  
        100 опыта = 1 уровень"""  
        return experience // 100
```

level_system_adapter.py

```
from ilevel_system import ILevelSystem
```

```
class LevelSystemAdapter(ILevelSystem):  
    """адаптер уровневой системы"""  
    def __init__(self, level_system):  
        self.level_system = level_system  
  
    def get_level_by_counter(self, counter):  
        """посчитать текущий уровень по количеству пройденных уровней  
        1 пройденное задание = 50 опыта  
        100 опыта = 1 уровень"""  
        experience = 50 * counter  
        return self.level_system.get_level_by_experience(experience)
```

imath.py:

```
from abc import ABC, abstractmethod
```

```
class IMath(ABC):  
    """интерфейс игры математика (сложение чисел)"""  
    @abstractmethod  
    def add(self):  
        """суммирование 2 чисел"""  
        pass
```

math_game.py:

```
from game import Game  
from random import randint  
from imath import IMath
```

```
MIN_NUM = 1  
MAX_NUM = 20
```

```
class MathGame(Game, IMath):  
    """игра - суммирование чисел (математика)"""  
    def __init__(self):  
        super().__init__()  
        self.__rand_num_1 = None  
        self.__rand_num_2 = None  
  
    def show_ask(self):  
        """показ вопроса"""  
        print(f'Вопрос №{self._counter + 1}\n\tЧему равно {self.__rand_num_1} + {self.__rand_num_2}?\')  
    def input_answer(self):  
        """ввод пользовательского ответа"""  
        num = input('Введите сумму чисел:\t')  
        self._user_answer = num  
  
    def generate_rand_nums(self):  
        """генерация двух случайных чисел для суммирования"""  
        self.__rand_num_1 = randint(MIN_NUM, MAX_NUM)  
        self.__rand_num_2 = randint(MIN_NUM, MAX_NUM)  
  
    def add(self):  
        """суммирование 2 чисел"""  
        self._correct_answer = str(self.__rand_num_1 + self.__rand_num_2)  
        return self._correct_answer
```

math_game_proxy.py:

```
from game import Game
from imath import IMath
from datetime import datetime
```

```
MATH_LOG_FILE = 'math_log.txt'
```

```
MIN_NUM = 1
MAX_NUM = 20
```

```
def log_add(filename):
    """логирование (хранение истории обращений)"""
    with open(filename, 'a') as file:
        cur_date = datetime.now()
        adding_log = f'adding; время: {cur_date}\n'
        file.write(adding_log)
```

```
class MathGameProxy(Game, IMath):
    """логирующий заместитель игры - суммирование чисел (математика)"""
    def __init__(self):
        super().__init__()
        self._math_game = None

    def set_original_math(self, math_game):
        """ленивая инициализация реального объекта"""
        if self._math_game is None:
            self._math_game = math_game

    def generate_rand_nums(self):
        """генерация двух случайных чисел для суммирования"""
        self._math_game.generate_rand_nums()

    def add(self):
        """суммирование 2 чисел"""
        self._math_game.add()
        log_add(MATH_LOG_FILE)

    def show_ask(self):
        """показать вопрос"""
        self._math_game.show_ask()

    def input_answer(self):
        """ввод пользовательского ответа"""
        self._math_game.input_answer()

    def check_correct_answer(self):
        """проверка правильности ответа пользователя"""
        return self._math_game.check_correct_answer()

    def show_message_about_answer(self, is_correct):
```

```
        """сообщение о правильности ответа"""
        self._math_game.show_message_about_answer(is_correct)
```

```
def go_to_next(self):
    """увеличение счетчика (переход на следующий уровень/вопрос)"""
    self._math_game.go_to_next()
```

quiz_file_reader.py:

```
from ifile_reader import IFileReader
from ifile_reader import IFileReader
import csv
```

```
class QuizFileReader(IFileReader):
    """читатель файла для игры викторина (читает вопрос, варианты ответа, и номер правильного
    ответа)"""
```

```
    def __init__(self, filename):
        self._filename = filename
```

```
    def get_filename(self):
        """getter filename"""
        return self._filename
```

```
    def get_concrete_data(self, index):
        """получать вопрос, номер правильного ответа и варианты ответа"""
```

```
    with open(self._filename, 'r', newline='') as File:
        reader = csv.reader(File, delimiter=',')
        counter = 0
        for ask_i, row in enumerate(reader):
            ask = row[0]
            correct_answer_num = next(reader)[0]
            answers = next(reader)
            counter += 1
            if ask_i == index:
                return [ask, correct_answer_num, answers]
        return None
```

quiz_file_reader_counter.py

```
from advanced_quiz_file_reader import AdvancedQuizFileReader
import csv

class QuizFileReaderCounter(AdvancedQuizFileReader):
    """читатель-декоратор файла для игры викторина
    (читает вопрос, варианты ответа, и номер правильного ответа)
    дополняет функционал за счет способности подсчитывать количество оставшихся в файле уровней"""
    def __init__(self, file_reader):
        self._file_reader = file_reader

    def get_concrete_data(self, index):
        """получать вопрос, номер правильного ответа и варианты ответа"""
        counter = self.get_remaining_data_num(index)
        print(f'Осталось вопросов: {counter}')
        return self._file_reader.get_concrete_data(index)

    def get_filename(self):
        """getter filename"""
        return self._file_reader.get_filename()

    def get_remaining_data_num(self, index):
        """подсчитывает количество оставшихся вопросов в файле"""
        with open(self._file_reader.get_filename(), 'r', newline='') as File:
            reader = csv.reader(File, delimiter=',')
            counter = 0
            for ask_i, row in enumerate(reader):
                next(reader)
                next(reader)
                if ask_i < index:
                    continue
                counter += 1
            return counter
```

quiz_game.py:

```
from game import Game
from quiz_file_reader import QuizFileReader
from quiz_file_reader_counter import QuizFileReaderCounter

QUIZ_ASK_FILE = 'quiz_asks.csv'

class QuizGame(Game):
    """игра - викторина (по русскому языку)"""
    def __init__(self):
        super().__init__()
        self._file_reader = QuizFileReaderCounter(QuizFileReader(QUIZ_ASK_FILE)) #
        QuizFileReader(QUIZ_ASK_FILE)

    def __init__(self, filename):
        super().__init__()
        self._file_reader = QuizFileReaderCounter(QuizFileReader(filename)) # QuizFileReader(filename)

    def show_ask(self):
        """показ вопроса и вариантов ответа текущего уровня
        concrete_data= [ask, num_correct_answer, answers]"""
        concrete_data = self._file_reader.get_concrete_data(self._counter)
        if concrete_data is None:
```

```

        print('Вопросы кончились!')
        return False
    ask = conrete_data[0]
    self._correct_answer = conrete_data[1]
    answers = conrete_data[2]
    print(f'Вопрос №{self._counter + 1}\n\t{ask}')
    for i, answer in enumerate(answers):
        print(f'\t\t{i + 1}) {answer}')
    return True

def input_answer(self):
    """ввод пользовательского ответа"""
    num = input('Введите номер правильного ответа:\t')
    self._user_answer = num

```

quiz_asks.csv:

Какая часть слова является носителем его значения?

2

приставка, корень, суффикс, окончание

По какому принципу слова объединяются в части речи?

2

по назначению, по общности грамматических свойств, по смыслу, по историческому принципу

Укажите падеж выделенного имени существительного. Под ДЕРЕВЬЯМИ зеленеют обросшие брусничником кочки. (Соколов-Микитов И.)

4

родительный, дательный, винительный, творительный

Какой род прилагательных не выделяют в русском языке?

4

мужской, средний, женский, общий

Определите род прилагательного, которое употреблено в предложении: «Солнце скрылось за темной тучей».

3

мужской, средний, женский, общий

В каком из словосочетаний употреблено прилагательное мужского рода?

1

летний дождь, длинная дорога, новое платье, сладкая ягода

math_log.txt:

adding; время: 2022-04-20 02:06:28.384450

adding; время: 2022-04-20 02:06:31.581248

adding; время: 2022-04-20 02:06:46.565661

adding; время: 2022-04-20 02:06:49.972286

adding; время: 2022-04-20 02:06:53.222613

adding; время: 2022-04-20 02:06:55.897448

adding; время: 2022-04-20 02:06:58.671007

adding; время: 2022-04-20 02:06:59.645661

adding; время: 2022-04-20 14:21:02.534213

main.py

```
from quiz_game import QuizGame
from math_game import MathGame
from math_game_proxy import MathGameProxy
```

```
QUIZ_ASK_FILE = 'quiz_asks.csv'
```

```
def main():
    print('Обучающий игровой комплекс для младших школьников')
    quiz_game = QuizGame(QUIZ_ASK_FILE)
    math_game_proxy = MathGameProxy()
    math_game = MathGame()
    math_game_proxy.set_original_math(math_game)
    while True:
        choice = input(f'\n{"-" * 100}\nСделайте выбор:\n\t1 - математика\n\t2 - викторина (по русскому
языку)'
                      f'\n\n\t0 - выйти\n:')
        if choice == '1':

            math_game_proxy.generate_rand_nums()
            math_game_proxy.add()
            math_game_proxy.show_ask()
            math_game_proxy.input_answer()
            is_correct = math_game_proxy.check_correct_answer()
            math_game_proxy.show_message_about_answer(is_correct)
            if is_correct:
                math_game_proxy.go_to_next()

        elif choice == '2':
            if quiz_game.show_ask():
                quiz_game.input_answer()
                is_correct = quiz_game.check_correct_answer()
                quiz_game.show_message_about_answer(is_correct)
                if is_correct:
                    quiz_game.go_to_next()
        elif choice == '3':
            pass
        elif choice == '0':
            return

if __name__ == '__main__':
    main()
```


Тесты:

Декоратор добавляет информацию о оставшихся вопросах, а адаптер позволяет высчитывать текущий уровень

```
C:\Python38-32\python.exe C:/
Users/Victor/MyFolder/AltS
TU_repo/6_sem/OOP_Kru
chkova-Savchenko/лаб5-6/
python_project_OOP_5-6/
main.py
```

Обучающий игровой
комплекс для младших
школьников

Сделайте выбор:

1 - математика

2 - викторина (по русскому
языку)

0 - выйти

:2

Осталось вопросов: 6

Вопрос №1

Какая часть слова является
носителем его значения?

1) приставка

2) корень

3) суффикс

4) окончание

Введите номер правильного
ответа: 2

Вы ответили правильно!
(ответ: 2)

Переход на следующее
задание!

Ваш текущий уровень 0!

Сделайте выбор:

1 - математика

2 - викторина (по русскому языку)

0 - выйти

:2

Осталось вопросов: 5

Вопрос №2

По какому принципу слова объединяются в части речи?

- 1) по назначению
- 2) по общности грамматических свойств
- 3) по смыслу
- 4) по историческому принципу

Введите номер правильного ответа: 2

Вы ответили правильно!
(ответ: 2)

Переход на следующее задание!

Ваш текущий уровень 1!

Сделайте выбор:

- 1 - математика
- 2 - викторина (по русскому языку)

0 - выйти

:2

Осталось вопросов: 4

Вопрос №3

Укажите падеж выделенного имени существительного. Под ДЕРЕВЬЯМИ зеленеют обросшие брусничником кочки. (Соколов-Микитов И.)

- 1) родительный

- 2) дательный
- 3) винительный
- 4) творительный

Введите номер правильного
ответа: 2

Вы ответили неверно!

Сделайте выбор:

- 1 - математика
- 2 - викторина (по русскому
языку)

0 - выйти

:2

Осталось вопросов: 4

Вопрос №3

Укажите падеж
выделенного имени
существительного. Под
ДЕРЕВЬЯМИ зеленеют
обросшие брусничником
кочки. (Соколов-Микитов
И.)

- 1) родительный
- 2) дательный
- 3) винительный
- 4) творительный

Введите номер правильного
ответа: 4

Вы ответили правильно!
(ответ: 4)

Переход на следующее
задание!

Ваш текущий уровень 1!

Сделайте выбор:

- 1 - математика

2 - викторина (по русскому
языку)

0 - выйти

:2

Осталось вопросов: 3

Вопрос №4

Какой род прилагательных
не выделяют в русском
языке?

- 1) мужской
- 2) средний
- 3) женский
- 4) общий

Введите номер правильного
ответа: 4

Вы ответили правильно!
(ответ: 4)

Переход на следующее
задание!

Ваш текущий уровень 2!

Сделайте выбор:

1 - математика

2 - викторина (по русскому
языку)

0 - выйти

:0

Process finished with exit code
0