

**Комплект оценочных материалов по дисциплине
«Объектно-ориентированные языки и системы программирования»**

Задания закрытого типа

Задания закрытого типа на выбор правильного ответа

1. Выберите один правильный ответ.

Какой паттерн лучше использовать для создания объекта, когда заранее неизвестен его конкретный тип?

- А) Singleton (Одиночка)
- Б) Builder (Строитель)
- В) Factory Method (Фабричный метод)
- Г) Prototype (Прототип)

Правильный ответ: В

Компетенции (индикаторы): ОПК-2

2. Выберите один правильный ответ.

Какие проблемы решает паттерн Decorator (Декоратор)?

- А) создание объектов
- Б) управление доступом
- В) обработка событий
- Г) расширение функциональности динамически

Правильный ответ: Г

Компетенции (индикаторы): ОПК-2

3. В чем разница между паттернами Strategy (Стратегия) и State (Состояние)?

Благодаря чему отслеживаются действия?

- А) Strategy проще в реализации
- Б) Strategy для алгоритмов, State для состояний
- В) State эффективнее
- Г) нет разницы

Правильный ответ: Б

Компетенции (индикаторы): ОПК-2

4. Когда стоит использовать паттерн Adapter (Адаптер)?

- А) для совместимости интерфейсов
- Б) для создания объектов
- В) для кэширования
- Г) для обработки событий

Правильный ответ: А

Компетенции (индикаторы): ОПК-2

Задания закрытого типа на установление соответствия

1. Установите правильное соответствие. Каждому элементу левого столбца соответствует только один правого столбца.

Установите соответствие между названиями и возможностями порождающих паттернов.

- | | |
|---|--|
| 1) Фабричный метод (Factory Method) | А) Гарантирует, что некоторый класс может иметь только один экземпляр, и предоставляет глобальную точку доступа к нему. |
| 2) Абстрактная фабрика (Abstract Factory) | Б) Предоставляет интерфейс для создания семейств, связанных между собой, или независимых объектов, конкретные классы которых неизвестны. |
| 3) Одиночка (Singleton) | В) Определяет интерфейс для создания объектов, при этом выбранный класс инстанцируется подклассами. |

Правильный ответ: 1-В, 2-Б, 3-А

Компетенции (индикаторы): ОПК-2

2. Установите правильное соответствие. Каждому элементу левого столбца соответствует только один правого столбца.

Установите соответствие между названиями и возможностями структурных паттернов.

- | | |
|----------------------------|--|
| 1) Decorator (декоратор) | А) Группирует объекты в древовидные структуры для представления иерархий типа «часть-целое». |
| 2) Composite (компоновщик) | Б) Подменяет другой объект для контроля доступа к нему. |
| 3) Bridge (мост) | В) Динамически возлагает на объект новые функции. |
| 4) Прoxy (заместитель) | Г) Отделяет абстракцию от реализации, благодаря чему появляется возможность независимо изменять то и другое. |

Правильный ответ: 1-В, 2-А, 3-Г, 4-Б

Компетенции (индикаторы): ОПК-2

3. Установите правильное соответствие. Каждому элементу левого столбца соответствует только один правого столбца.

Установите соответствие между названиями и возможностями паттернов поведения.

- | | |
|-------------------------|---|
| 1) Mediator (посредник) | А) Дает возможность последовательно обойти все элементы составного объекта, не раскрывая его внутреннего представления. |
| 2) Iterator (итератор) | Б) является базовым для приложений и не содержит классов, относящихся к интерфейсу пользователя. |
| 3) Interpreter | В) Для заданного языка определяет представление |

(интерпретатор) его грамматики, а также интерпретатор предложений языка, использующий это представление.

Правильный ответ: 1-В, 2-А, 3-Б

Компетенции (индикаторы): ОПК-2

Задания закрытого типа на установление правильной последовательности

1. Установите правильную последовательность. Запишите правильную последовательность букв слева направо.

Восстановите порядок написания программы объектно-ориентированного программирования (ООП) на языке C++:

А) создание классов объектов

Б) объявление объектов данных классов

В) определение задачи

Г) определение сообщений, принимаемых каждым объектом, и кодов функций, согласно которым объект будет реагировать на эти сообщения

Д) компиляция и компоновка системы

Е) определение начального состояния системы

Правильный ответ: В, А, Г, Б, Е, Д

Компетенции (индикаторы): ОПК-2

2. Установите правильную последовательность. Запишите правильную последовательность букв слева направо.

Укажите правильную последовательность объектов MVC:

А) вид

Б) модель

В) контроллер

Правильный ответ: Б, А, В

Компетенции (индикаторы): ОПК-2

3. Установите правильную последовательность. Запишите правильную последовательность букв слева направо.

Расположите этапы жизненного цикла разработки ПО в порядке их выполнения:

А) Проектирование архитектуры

Б) Реализация (кодирование)

В) Дизайн и моделирование

Г) Сбор и анализ требований

Д) Поддержка и сопровождение

Е) Тестирование

Ж) Вывод из эксплуатации

З) Внедрение

Правильный ответ: Г, А, В, Б, Е, З, Д, Ж

Компетенции (индикаторы): ОПК-2

Задания открытого типа

Задания открытого типа на дополнение

1. Напишите пропущенное слово (словосочетание)

Под паттернами _____ понимается описание взаимодействия объектов и классов, адаптированных для решения общей задачи проектирования в конкретном контексте.

Правильный ответ: проектирования

Компетенции (индикаторы): ОПК-2

2. Напишите пропущенное слово (словосочетание)

_____ паттерны связаны с процессом создания объектов.

Правильный ответ: порождающие

Компетенции (индикаторы): ОПК-2

3. Напишите пропущенное слово (словосочетание)

_____ паттерны имеют отношение к композиции объектов и классов.

Правильный ответ: структурные

Компетенции (индикаторы): ОПК-2

4. Напишите пропущенное слово (словосочетание)

Паттерны _____ характеризуют то, как классы или объекты взаимодействуют между собой.

Правильный ответ: поведения

Компетенции (индикаторы): ОПК-2

5. Напишите пропущенное слово (словосочетание)

_____ - это набор взаимодействующих классов, составляющих повторно используемый дизайн для конкретного класса программ.

Правильный ответ: каркас

Компетенции (индикаторы): ОПК-2

Задания открытого типа с кратким свободным ответом

1. _____ преобразует интерфейс класса в некоторый другой интерфейс, ожидаемый клиентами, обеспечивает совместную работу классов, которая была бы невозможна без данного паттерна из-за несовместимости интерфейсов.

Правильный ответ: adapter / адаптер

Компетенции (индикаторы): ОПК-2

2. _____ отделяет конструирование сложного объекта от его представления, позволяя использовать один и тот же процесс конструирования для создания различных представлений.

Правильный ответ: builder / строитель

Компетенции (индикаторы): ОПК-2

3. _____ инкапсулирует запрос в виде объекта, позволяя тем самым параметризовывать клиентов типом запроса, устанавливать очередность запросов, протоколировать их и поддерживать отмену выполнения операций.

Правильный ответ: command / команда

Компетенции (индикаторы): ОПК-2

4. _____ предоставляет унифицированный интерфейс к множеству интерфейсов в некоторой подсистеме, определяет интерфейс более высокого уровня, облегчающий работу с подсистемой.

Правильный ответ: facade / фасад

Компетенции (индикаторы): ОПК-2

5. _____ позволяет, не нарушая инкапсуляции, получить и сохранить во внешней памяти внутреннее состояние объекта, чтобы позже объект можно было восстановить точно в таком же состоянии.

Правильный ответ: memento / хранитель

Компетенции (индикаторы): ОПК-2

6. _____ определяет между объектами зависимость типа один-ко-многим, так что при изменении состояния одного объекта все зависящие от него получают извещение и автоматически обновляются.

Правильный ответ: observer / наблюдатель

Компетенции (индикаторы): ОПК-2

7. _____ описывает виды создаваемых объектов с помощью прототипа и создает новые объекты путем его копирования.

Правильный ответ: prototype / прототип

Компетенции (индикаторы): ОПК-2

Задания открытого типа с развернутым ответом

1. Разработать приложение для системы логистики, в которой задействованы разные типы транспорта - грузовики и корабли, обеспечить добавление нового типа транспорта (самолетов).

Компоненты:

1) Продукт (Product): Определяет общий интерфейс для всех создаваемых объектов.

2) Конкретный продукт (Concrete Product): Реализует интерфейс продукта.

3) Создатель (Creator): Определяет фабричный метод, который возвращает объекты продукта.

4) Конкретный создатель (Concrete Creator): Реализует фабричный метод, создавая определённый тип продукта.

Время выполнения – 60 минут

Критерии оценивания: содержательное соответствие приведенному ниже коду:

```
#include <iostream>
#include <memory>
#include <string>
// Продукт
class Transport {
public:
    virtual ~Transport() = default;
    virtual void deliver() const = 0;
};
// Конкретные продукты
class Truck : public Transport {
public:
    void deliver() const override {
        std::cout << "Доставка по суше грузовиком."
<< std::endl;
    }
};
class Ship : public Transport {
public:
    void deliver() const override {
        std::cout << "Доставка по морю кораблем."
<< std::endl;
    }
};
// Создатель
class Logistics {
public:
```

```

        virtual ~Logistics() = default;
        virtual std::unique_ptr<Transport>
createTransport() const = 0;
        void planDelivery() const {
            auto transport = createTransport();
            transport->deliver();
        }
    };
    // Конкретные создатели
    class RoadLogistics : public Logistics {
    public:
        std::unique_ptr<Transport> createTransport()
const override {
            return std::make_unique<Truck>();
        }
    };
    class SeaLogistics : public Logistics {
    public:
        std::unique_ptr<Transport> createTransport()
const override {
            return std::make_unique<Ship>();
        }
    };
    // Клиентский код
    int main() {
        std::unique_ptr<Logistics> logistics =
std::make_unique<RoadLogistics>();
        logistics->planDelivery();
        logistics = std::make_unique<SeaLogistics>();
        logistics->planDelivery();
        return 0;
    }

```

Компетенции (индикаторы): ОПК-2

2. Разработать приложение: система управления платёжными сервисами уже работает с одним API, нужно интегрировать сторонний платёжный сервис с другим интерфейсом.

Компоненты:

- 1) Клиент (Client): Класс, использующий целевой интерфейс.
- 2) Целевой интерфейс (Target): Интерфейс, ожидаемый клиентом.
- 3) Адаптируемый класс (Adaptee): Класс с несовместимым интерфейсом, который нужно адаптировать.
- 4) Адаптер (Adapter): Класс, который реализует целевой интерфейс и использует адаптируемый класс для выполнения своей работы.

Время выполнения – 60 минут

Критерии оценивания: содержательное соответствие приведенному ниже коду:

```
#include <iostream>
#include <string>
// Целевой интерфейс (Target)
class PaymentProcessor {
public:
    virtual ~PaymentProcessor() = default;
    virtual void processPayment(double amount)
const = 0;
};
// Адаптируемый класс (Adaptee)
class OldPaymentSystem {
public:
    void makeTransaction(const std::string&
details) const {
        std::cout << "Processing payment with
details: " << details << std::endl;
    }
};
// Адаптер (Adapter)
class PaymentAdapter : public PaymentProcessor {
private:
    OldPaymentSystem* adaptee;
public:
    PaymentAdapter(OldPaymentSystem* oldSystem) :
adaptee(oldSystem) {}
    void processPayment(double amount) const
override {
        // Преобразуем данные в формат, понятный
старой системе
        std::string details = "Amount: " +
std::to_string(amount);
        adaptee->makeTransaction(details);
    }
};
// Клиент (Client)
class OnlineStore {
private:
    const PaymentProcessor* paymentProcessor;
public:
    OnlineStore(const PaymentProcessor* processor)
: paymentProcessor(processor) {}
    void checkout(double amount) const {
```

```

        std::cout << "Starting payment
process...\n";
        paymentProcessor->processPayment(amount);
        std::cout << "Payment complete.\n";
    }
};
// Клиентский код
int main() {
    OldPaymentSystem oldSystem;
    PaymentAdapter adapter(&oldSystem);
    OnlineStore store(&adapter);
    store.checkout(99.99);
    return 0;
}

```

Компетенции (индикаторы): ОПК-2

Экспертное заключение

Представленный комплект оценочных материалов по дисциплине «Объектно-ориентированные языки и системы программирования» соответствует требованиям ФГОС ВО.

Предлагаемые оценочные материалы адекватны целям и задачам реализации основной профессиональной образовательной программы по направлению подготовки 09.04.04 Программная инженерия.

Виды оценочных средств, включенные в представленный фонд, отвечают основным принципам формирования ФОС.

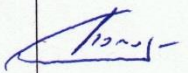
Разработанные и представленные для экспертизы оценочные материалы рекомендуются к использованию в процессе подготовки обучающихся по указанному направлению.

Председатель учебно-методической комиссии
института компьютерных систем
и информационных технологий



Ветрова Н.Н.

Лист изменений и дополнений

№ п/п	Виды дополнений и изменений	Дата и номер протокола заседания кафедры (кафедр), на котором были рассмотрены и одобрены изменения и дополнения	Подпись (с расшифровкой) заведующего кафедрой (заведующих кафедрами)
1.	Дополнен комплект оценочных материалов	протокол заседания кафедры компьютерных систем и сетей № <u>8</u> от <u>10.03.2025</u>	 С.В. Попов