

МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Ухтинский государственный технический университет»
(УГТУ)
филиал Ухтинского государственного технического университета
в г. Усинске
(УФ УГТУ)
(среднего профессионального образования)

УТВЕРЖДАЮ
И. о. директора филиала
_____ О. В. Филиппова
« ____ » _____ 20 ____ г.

КОМПЛЕКС ОЦЕНОЧНЫХ СРЕДСТВ
ПО УЧЕБНОМУ МОДУЛЮ
ПМ.02 РАЗРАБОТКА И ИНТЕГРАЦИЯ МОДУЛЕЙ ПО
УП.02.01 Учебная практика
образовательной программы
среднего профессионального образования
По специальности 09.02.11 Разработка и управление программным
обеспечением

г. Усинск
2026

1. ПАСПОРТ КОМПЛЕКТА ОЦЕНОЧНЫХ СРЕДСТВ

1.1 Область применения

Комплект оценочных средств (далее – КОС) предназначен для контроля и оценки результатов прохождения учебной практики УП.02.01 Учебная практика, образовательной программы среднего профессионального образования (далее – ОП СПО) по специальности 09.02.11 Разработка и управление программным обеспечением.

1.2 Результаты освоения компетенции

В результате проведения итоговой аттестации по учебной практике комплексная оценка овладения следующими профессиональными и общими компетенциями:

Код	Результат освоения компетенции
ПК 2.1	Проектировать модули программного обеспечения.
ПК 2.2	Разрабатывать модули программного обеспечения.
ПК 2.3	Выполнять интеграцию модулей и компонентов программного обеспечения.
ПК 2.4	Выполнять тестирование и отладку программного обеспечения
ПК 2.5	Осуществлять документирование программных модулей программного обеспечения.
ОК 1	Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам
ОК 2	Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности
ОК 3	Планировать и реализовывать собственное профессиональное и личностное развитие, предпринимательскую деятельность в профессиональной сфере, использовать знания по правовой и финансовой грамотности в различных жизненных ситуациях
ОК 4	Эффективно взаимодействовать и работать в коллективе и команде
ОК 5	Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста
ОК 7	Содействовать сохранению окружающей среды, ресурсосбережению, применять знания об изменении климата, принципы бережливого производства, эффективно действовать в чрезвычайных ситуациях

ОК 9	Пользоваться профессиональной документацией на государственном и иностранном языках
-------------	---

2. ФОРМЫ КОНТРОЛЯ И ОЦЕНКИ РЕЗУЛЬТАТОВ ОСВОЕНИЯ УЧЕБНОЙ ДИСЦИПЛИНЫ

В соответствии с учебным планом и рабочей программой предусматривается текущий и промежуточный контроль результатов освоения.

Предметом оценки служат умения и знания, предусмотренные требованиями ФГОС по практике УП.02.01 Учебная практика, направленные на формирование общих и профессиональных компетенций, а также личностных результатов в рамках программы воспитания.

2.1 Критерии и нормы оценки знаний обучающихся.

Оценка «зачтено» выставляется, если студент твёрдо знает материал, грамотно и по существу излагает его, не допуская существенных неточностей в ответе на вопрос, правильно применяет теоретические положения при решении практических вопросов и задач, владеет необходимыми знаниями, умениями и навыками.

Оценка «не зачтено» выставляется, если студент не знает значительной части программного материала, допускает существенные ошибки, неуверенно, с большими затруднениями выполняет практические работы.

Также при оценке зачёта могут учитываться:

- Степень усвоения теоретического материала. Оценивается по правильности ответа на теоретический вопрос, последовательности, связанности и чёткости изложения материала, объёму теоретических знаний в рамках программного материала, умению выделить главные положения в изученном материале, культуре речи.
- Уровень сформированности умений и навыков. Оценивается по правильности реализации алгоритма решения практической задачи, правильности интерпретации полученных результатов, умению сделать выводы из полученных значений.

2.3 Формы и методы оценивания производственной практики ПП.04.01

Форма аттестации - **зачет**

Условия аттестации: аттестация проводится в устной форме по завершению освоения учебной практики при положительных результатах.

Время аттестации: Подготовка - 45 мин.; устный ответ — 10 мин.

Зачет по учебной практике выставляется на основании данных аттестационного листа (характеристики профессиональной деятельности обучающегося на практике) с указанием видов работ, выполненных обучающимися во время практики, их объема, качества выполнения.

3. Контрольно-оценочные материалы для итоговой аттестации по учебной практике УП.02.01

ВОПРОСЫ ДЛЯ ПОДГОТОВКИ К ЗАЧЕТУ ПО УЧЕБНОЙ ПРАКТИКЕ УП.02.01

3.1. Примерный перечень вопросов к зачету.

Часть 1. Методы и этапы технологии программирования.

1. Какая модель построения программ лежит в основе технологии процедурного программирования? Каковы основные методы процедурного программирования?

Ответ:

Процедурное программирование основано на модели построения программы как иерархии процедур, что и дало название данной технологии.

Технология процедурного программирования основана на использовании следующих методов (приемов) программирования:

- 1) метод декомпозиции (нисходящего проектирования), т. е. разделение программы на процедуры простейшей структуры и представление программы в виде иерархии процедур;
- 2) метод модульной организации, т. е. группировка процедур и обрабатываемых ими данных в модули, которые программируются и компилируются отдельно. Преимущества данного метода заключаются в параллельной работе программистов, удобстве программирования, возможности создания библиотек;
- 3) метод структурного программирования процедур, который заключается в следующем:
 - разделение процедур на вложенные блоки, что позволяет локализовать переменные и операторы их обработки, структурировать процедуру;
 - использование операторов ветвления и циклов, осуществляющих передачу управления только сверху — вниз, что приводит к ясности алгоритма, к облегчению программирования и сопровождения программ; операторы безусловной передачи управления goto использовать не рекомендуется;
 - форматирование текста процедуры, т. е. использование отступов для отображения вложенности блоков, применение идентификаторов, несущих смысл, и использование комментариев, что приводит к повышению читаемости программ и к облегчению их сопровождения.

2. На чем основывается концепция объектно-ориентированного программирования? Назовите основные принципы объектно-ориентированного программирования.

Ответ:

Концепция объектно-ориентированного программирования (ООП) основывается на концепции объектов. Программа строится как набор объектов, каждый из которых — экземпляр определённого класса. Объекты объединяют данные (состояние) и методы (поведение), которые работают с этими данными, в единое целое.

Четыре основных принципа объектно-ориентированного программирования (ООП):

1. Инкапсуляция. Объект скрывает внутреннее устройство и предоставляет только необходимый внешний интерфейс. Это уменьшает связанность компонентов и упрощает сопровождение.
2. Наследование. Один класс (объект-шаблон) может наследовать свойства и поведение другого. Это позволяет создавать более специализированные классы на основе общих.
3. Полиморфизм. Разные объекты могут реализовывать одинаковые методы по-разному. Это упрощает работу с обобщёнными интерфейсами и расширение функциональности.
4. Абстракция. Сложные системы представляются в виде простых моделей, скрывающих ненужные детали. Программист оперирует понятиями «что делает», а не «как именно».

3. Что такое компонентные технологии и CASE-технологии?

Ответ:

Компонентные технологии — подход, при котором программное обеспечение строится из заранее созданных и протестированных компонентов.

CASE-технологии (Computer-Aided Software/System Engineering) — инструментальные средства, используемые при проектировании систем, охватывающие весь спектр работ по созданию и сопровождению ПО (анализ и разработка, составление проектной документации, кодирование и тестирование системы).

4. Что такое жизненный цикл программного обеспечения? Какие модели жизненного цикла программного обеспечения вы знаете?

Ответ:

Жизненный цикл программного обеспечения (ПО) — период времени, который начинается с момента принятия решения о необходимости создания программного продукта и заканчивается в момент его полного изъятия из эксплуатации.

Среди моделей жизненного цикла программного обеспечения (ПО) известны каскадная (водопадная), инкрементная и спиральная.

1. Каскадная (водопад)

Предполагает последовательное выполнение этапов. Каждый этап следует строго один за другим, без возврата на предыдущие стадии. Разработка идёт «каскадом» сверху вниз, подобно водопаду.

2. Инкрементная

Предполагает поэтапное добавление новой функциональности на каждом этапе до достижения конечного продукта. Каждый этап (инкремент) представляет собой

полноценную работающую версию продукта с определённым набором функциональных возможностей.

3. Спиральная

Использует циклический подход к разработке. Процесс проходит через несколько итераций, каждая из которых завершается оценкой рисков и принятием решения о дальнейшей работе. Каждая итерация включает в себя анализ требований, проектирование, разработку, тестирование и оценку рисков.

5. Что называется архитектурой программного обеспечения?

Ответ:

Архитектура программного обеспечения (ПО) — это высокоуровневая структура системы, определяющая её компоненты, их взаимодействие, правила и руководящие принципы, регулирующие дизайн на протяжении всего жизненного цикла приложения.

Архитектура охватывает как функциональные, так и нефункциональные требования: безопасность, производительность, масштабируемость, удобство сопровождения.

6. Какими свойствами обладают алгоритмы? Какие существуют формы записи алгоритма?

Ответ:

Основные свойства алгоритмов:

- определенность (детерминированность) — точность и понятность, обеспечивающие однозначность результата;
- результативность — получение искомого результата через конечное число этапов (шагов);
- дискретность — расчлененность алгоритма на отдельные этапы (шаги);
- массовость — пригодность алгоритма для решения всех задач данного типа.

Существуют следующие способы записи алгоритмов:

- псевдокод (формульно-словесный) — содержание последовательных этапов вычислений задается с нумерацией в произвольной форме на естественном языке; это искусственный, неформальный язык, но не язык программирования;
- блок-схемный — изображение структуры алгоритма в виде графического представления этапов процесса обработки данных с помощью специальных геометрических символов (блоков).
- алгоритмические языки — непосредственная запись алгоритма решения задачи с помощью операторов языка.

7. Каковы основные типы пользовательских интерфейсов?

Ответ:

Пользовательский интерфейс представляет собой совокупность программных и аппаратных средств, обеспечивающих диалог пользователя и компьютера.

Диалог — это процесс обмена информацией между пользователем и программой, осуществляемый в реальном масштабе времени и служащий для решения конкретной задачи.

Различают следующие типы пользовательских интерфейсов:

- примитивные интерфейсы;
- интерфейсы-меню;
- интерфейсы со свободной навигацией;
- интерфейсы прямого манипулирования.

Примитивный интерфейс реализует единственный сценарий работы программного обеспечения: например, ввод данных — обработка — вывод результатов. Подобный интерфейс применяется крайне редко, например, когда программа имеет только одну функцию.

Интерфейс-меню реализует множество сценариев работы с иерархической структурой команд, выбираемых пользователем. Иерархическая организация меню с графом типа «дерева» задает строго ограниченную навигацию: вверх-вниз, вправо-влево по ветвям графа.

Интерфейс со свободной навигацией реализует множество сценариев работы, не привязанных к уровням иерархии команд. Такой интерфейс предполагает определение множества команд на конкретном шаге работы.

Преимущества интерфейсов со свободной навигацией по сравнению с интерфейсами-меню заключаются в возможности доступа к любым командам и в выборе команд, имеющих смысл в данный момент. Диалоговые окна программы содержат меню различных видов (кнопочное, ниспадающее, контекстное) и элементы управления вводом/выводом, реализующие диалог с пользователем.

Интерфейс прямого манипулирования реализует множество сценариев работы посредством выбора и перемещения пиктограмм, определяющих объекты предметной области. Этот тип интерфейса реализован в интерфейсе операционной системы Windows, является альтернативой интерфейсу со свободной навигацией.

Часть 2. Программирование на языке высокого уровня (Python).

1. Что выполняют команды `print()` и `input()`? Как влияет на команду `print()` параметры `sep` и `end`?

Ответ:

Для вывода данных на экран используется команда `print()`.

Внутри круглых скобок пишем, что хотим вывести на экран. Если это текст, то обязательно нужно заключить его внутри кавычек. Кавычки могут быть одинарными (') или двойными ("). До и после текста необходимо ставить только одинаковые кавычки. Команда `print()` позволяет указывать несколько аргументов, в таком случае их надо отделять запятыми.

Команда `input()` считывает данные, введенные с клавиатуры.

Параметр `sep` команды `print()` позволяет установить строку, с помощью которой будут разделены аргументы (если их несколько) при печати.

Параметр `end` определяет строку, которая будет добавлена после вывода всех аргументов команды `print()`.

2. Условный оператор. Чем отличается вложенные от каскадных операторов.

Ответ:

Условный оператор в Python — это конструкция, которая позволяет выполнять различные блоки кода в зависимости от истинности (True) или ложности (False) определённого условия. В языке используются три основных условных оператора: if, elif и else.

Вложенные условные операторы:

Проверяют условия внутри других условий. Это полезно для обработки сложных логических сценариев, где необходимо учитывать несколько уровней условий.

Пример программного кода определения координатной четверти точки:

```
x = int(input())
y = int(input())

if x > 0:
    if y > 0:
        print('Первая четверть')
    else:
        print('Четвертая четверть')
else:
    if y > 0:
        print('Вторая четверть')
    else:
        print('Третья четверть')
```

Каскадные условные операторы:

Проверяют несколько условий последовательно и выполняют разные блоки кода в зависимости от результата проверки. Для этого используется конструкция if, elif и else.

Пример программного кода:

```
grade = int(input('Введите вашу отметку: '))

if grade >= 90:
    print(5)
elif grade >= 80:
    print(4)
elif grade >= 70:
    print(3)
elif grade >= 60:
    print(2)
else:
    print(1)
```

3. Цикл for. Какие параметры используются в функции range().

Ответ:

Цикл for — это конструкция, которая выполняет действия определённое количество раз. Её используют, когда заранее известно, сколько раз нужно повторить шаг.

Цикл подходит для перебора элементов в списках, строках и массивах, помогает обходить коллекции, выполняя нужные операции для каждого элемента.

Рассмотрим программный код:

```
for i in range(10):
```

```
    print('Привет', i)
```

Значение, которое мы указываем в скобках у функции range() обозначает количество итераций цикла, при этом переменная i принимает последовательно значения: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Перегрузка range() с двумя параметрами.

Если мы хотим начинать последовательность не с 0, а с какого-то другого числа, то мы можем использовать перегрузку функции range() принимающую два параметра. Например, вызов функции range(1, 5) сгенерирует последовательность чисел 1, 2, 3, 4 (правая граница не включительна).

Перегрузка range() с тремя параметрами

Первый параметр задает старт последовательности, второй параметр задает стоп последовательности и третий – шаг генерации чисел. Например, вызов функции range(1, 10, 2) создаст последовательность чисел 1, 3, 5, 7, 9, а вызов функции range(5, 30, 5) сгенерирует последовательность 5, 10, 15, 20, 25.

Если шаг генерации является положительным числом, то генерируемая последовательность будет возрастать. Мы можем указать отрицательный шаг генерации (третий параметр), что приведет к генерированию убывающей последовательности.

МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Ухтинский государственный технический университет»
(УГТУ)
филиал Ухтинского государственного технического университета
в г. Усинске
(УФ УГТУ)
(среднего профессионального образования)

УТВЕРЖДАЮ
И. о. директора филиала
О.В. Филиппова
«20» 2026 г.

(подпись) (И. О. Фамилия)
« » 20 г.

(подпись) (И. О. Фамилия)
« » 20 г.

РАБОЧАЯ ПРОГРАММА

Дисциплина: Учебная практика

Индекс: УП.02.01

09.02.11 Разработка и управление программным
Специальность: обеспечением

Форма обуче- очная
ния:

Курс (ы) 2

Семестр (ы): 3

г. Усинск
2026

Рабочая программа составлена в соответствии с требованиями Федерального государственного образовательного стандарта среднего общего образования, утвержденного приказом Минпросвещения России от 24.02.2025 № 138 "Об утверждении федерального государственного образовательного стандарта среднего профессионального образования по специальности 09.02.11 Разработка и управление программным обеспечением".

Разработчик Капелюшва И.А. преподаватель СПО УФ УГТУ.

Рассмотрено на заседании ученого совета УФ УГТУ (протокол от 26.05.2026 № 06).

СОГЛАСОВАНО

Заместитель директора УФ УГТУ
по учебной работе



О.В. Филиппова