

МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Документ подписан простой электронной подписью
Информация о владельце:

ФИО: Кислова Наталья Николаевна

Должность: Проректор по ИТ и качеству образования

Дата подписания: 01.09.2026 01:49:40

Уникальный программный ключ:

52802513f5b14a975b3e9b13008093d5726b159bf6064f865ae65b96a966c035

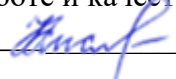
федеральное государственное бюджетное образовательное учреждение высшего образования

«Самарский государственный социально-педагогический университет»

Кафедра информатики, прикладной математики и методики их преподавания

Утверждаю

Проректор по учебно-методической
работе и качеству образования

 Н.Н. Кислова

Пугач Ольга Исааковна, Петрунин Дмитрий Андреевич

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

для проведения промежуточной аттестации по дисциплине
«Программирование»

Направление подготовки:

44.03.01 Педагогическое образование

Направленность (профиль):

«Информатика»

Квалификация выпускника

Бакалавр

Рассмотрено

Протокол №11 от 26.06.2026.

Заседания кафедры информатики, прикладной
математики и методики их преподавания

Одобрено

Начальник

Управления

образовательных программ

 Н.А. Доманина

Самара 2026

Пояснительная записка

Фонд оценочных средств (далее – ФОС) для промежуточной аттестации по дисциплине «Программирование» разработан в соответствии с федеральным государственным образовательным стандартом высшего образования – бакалавриат по направлению подготовки 44.03.05 Педагогическое образование (с двумя профилями подготовки), утвержденного приказом Министерства образования и науки Российской Федерации от 22 февраля 2018 г. № 125, основной профессиональной образовательной программой «Информатика» и «Фиджитал-спорт» с учетом требований профессионального стандарта 01.001 «Педагог (педагогическая деятельность в сфере дошкольного, начального общего, основного общего, среднего общего образования) (воспитатель, учитель)», утвержденный приказом Министерства труда и социальной защиты Российской Федерации от 18 октября 2013 г. № 544н (зарегистрирован Министерством юстиции Российской Федерации 6 декабря 2013 г., регистрационный № 30550), с изменениями, внесенными приказами Министерства труда и социальной защиты Российской Федерации от 25 декабря 2014 г. № 1115н (зарегистрирован Министерством юстиции Российской Федерации 19 февраля 2015 г., регистрационный № 36091) и от 5 августа 2016 г. № 422н (зарегистрирован Министерством юстиции Российской Федерации 23 августа 2016 г., регистрационный № 43326), 01.003 «Педагог дополнительного образования детей и взрослых» утвержденный приказом Министерства труда и социальной защиты Российской Федерации от 22 сентября 2012 г. № 652н от 22.09.2021 г. (Зарегистрировано в Минюсте России 17.12.2021 N 66403).

Цель ФОС для промежуточной аттестации – установление уровня сформированности части компетенции – ПК-1.

Задачи ФОС для промежуточной аттестации - контроль качества и уровня достижения результатов обучения по формируемым в соответствии с учебным планом компетенциям:

ПК-1. Способен осуществлять педагогическую деятельность по реализации образовательного процесса по предмету

ПК-1.1. Умеет реализовывать образовательную программу по предмету с учетом специфики содержания, методов и инструментов соответствующей области научного знания

Знает: современные методы процедурного, объектно-ориентированного, функционального программирования; основные конструкции языков программирования; основные типы данных и операторы.

Умеет: записывать разрабатываемые алгоритмы на языках программирования высокого уровня; комментировать синтаксические и семантические ошибки, возникающие при некорректном выполнении программы; отлаживать и тестировать задачи; составлять систему тестов для автоматизированной проверки корректности программы; подбирать оптимальное программное обеспечение для разработки, тестирования и отладки программ.

ПК-1.2. Реализует образовательную программу по предмету с использованием технологий профессиональной деятельности

Умеет: использовать знания по программированию при реализации образовательной программы по информатике.

Требование к процедуре оценки:

Помещение: особых требований нет

Расходные материалы: лист бумаги, ручка

Доступ к дополнительным справочным материалам: не предусмотрен

Нормы времени: 130 мин

Комплект оценочных средств для проведения промежуточной аттестации в 7 семестре

Раздел 1. Процедурное программирование на C++

Проверяемая компетенция:

ПК-1. Способен осуществлять педагогическую деятельность по реализации образовательного процесса по предмету
Проверяемый индикатор достижения компетенции:

ПК-1.1. Умеет реализовывать образовательную программу по предмету с учетом специфики содержания, методов и инструментов соответствующей области научного знания

Проверяемые результаты обучения:

Знает: современные методы процедурного, объектно-ориентированного, функционального программирования; основные конструкции языков программирования; основные типы данных и операторы.

Умеет: записывать разрабатываемые алгоритмы на языках программирования высокого уровня; комментировать синтаксические и семантические ошибки, возникающие при некорректном выполнении программы; отлаживать и тестировать задачи; составлять систему тестов для автоматизированной проверки корректности программы; подбирать оптимальное программное обеспечение для разработки, тестирования и отладки программ.

Проверяемый индикатор достижения компетенции:

ПК-1.2. Реализует образовательную программу по предмету с использованием технологий профессиональной деятельности

Проверяемые результаты обучения:

Умеет: использовать знания по программированию при реализации образовательной программы по информатике.

Тип (форма) задания: тестовые задания с выбором одного или нескольких вариантов ответа (А), практические задания (В).

Содержание задания:

A1. Выберите правильный вариант объявления константной переменной в C++, где type - тип данных в C++
variable - имя переменной value - константное значение

- a) const variable = value;
- b) const type variable = value;
- c) const type variable := value;

A2. Чтобы подключить заголовочный файл в программу на C++, например iostream необходимо написать:

- a) #include <> c iostream внутри скобок
- b) #include <>; c iostream.h внутри скобок
- c) include #iostream,h;
- d) include (iostreamh)

A3. Какой из ниже перечисленных операторов, не является циклом в C++?

- a) while
- b) repeat until
- c) for
- d) do while

A4. Чему будет равна переменная a, после выполнения этого кода int a; for(a = 0; a < 10; a++) {}?

- a) 9
- b) 10
- c) 1
- d) 17

A5. Укажите объектно-ориентированный язык программирования

- a) Все варианты ответов
- b) C++
- c) Java
- d) Eiffel

A6. Какую функцию должны содержать все программы на C++?

- a) system()
- b) start()
- c) main()
- d) program()

A7. Какие служебные символы используются для обозначения начала и конца блока кода?

- a) { }
- b) < >
- c) begin end
- d) ()

A8. Цикл с постусловием?

- a) while
- b) for
- c) do while
- d) repeat

A9. Какая из следующих записей - правильный комментарий в C++?

- a) /* Комментарий */
- b) {комментарий}
- c) /* комментарий */
- d) ** Комментарий **

A10. Программа, переводящая входную программу на исходном языке в эквивалентную ей выходную программу на результирующем языке, называется:

- a) интерпретатор
- b) сканер
- c) транслятор
- d) компилятор

A11. Цикл с предусловием?

- a) while
- b) do while
- c) for
- d) repeat

A12. Какой оператор не допускает перехода от одного константного выражения к другому?

- a) Stop;
- b) end;
- c) точка с запятой
- d) break;

A13. Какими знаками заканчивается большинство строк кода в Си++?

- a) ; (точка с запятой)
- b) : (двоеточие)
- c) , (запятая)
- d) . (точка)

A14. Каков результат работы следующего фрагмента кода?

```
int x = 0;
```

```
switch(x)
{
    case 1: cout << "Один";

    case 0: cout << "Ноль";

    case 2: cout << "Привет мир";
}
```

- a) НольПривет мир
- b) Привет мир
- c) Ноль
- d) Один

A15. Какое значение, по умолчанию, возвращает программа операционной системе в случае успешного завершения?

- a) 0
- b) 1
- c) -1
- d) Программа не возвращает значение.

A16. Простые типы данных в C++.

- a) целые – int, вещественные – float или double, символьные – string
- b) целые – int, вещественные – float или double, символьные – char
- c) целые – int, вещественные – float или real, символьные – char
- d) целые – bool, вещественные – float или double, символьные – string

A17. Какой служебный знак ставится после оператора case ?

- a) :
- b) ;
- c) .
- d) -

18. Укажите правильное определение функции main в соответствии со спецификацией стандарта ANSI

- a) int main(void)
- b) int main()
- c) void main()
- d) void main(void)

19. Какому зарезервированному слову программа передаёт управление в случае, если значение переменной или выражения оператора switch не совпадает ни с одним константным выражением?

- a) default
- b) all
- c) other
- d) contingency

20. Какие среды программирования (IDE) предназначены для разработки программных средств?

- a) MVS, NetBeans, QT Creator, RAD Studio, Dev-C++
- b) MVS, Code::Blocks, QT Creator, RAD Studio, MathCAD
- c) MVS, Code::Blocks, QT Creator, AutoCAD, Eclipse

Оценочный лист к типовому заданию А

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
a	a	b	a	a	c	a	c	c	c
A11	A12	A13	A14	A15	A16	A17	A18	A19	A20
a	d	a	a	a	b	a	a	a	a

Каждый верный ответ оценивается в 2 балла.

Задание В

B1. Оцените следующий алгоритм. Что вернет программа при вводе num5. Опишите, что делает программа в общем случае? Предложите альтернативный вариант решения.

```
#include <iostream>
#include <locale.h>
```

```
using namespace std;
```

```
int main()
{
    int number;
    setlocale(LC_CTYPE, "Russian");
    cout << "Введите число: ";
    cin >> number;
    cin.ignore();
    cout << "Вы ввели: "<< number << "\n";
    cin.get();
}
```

B2. Последовательность состоит из натуральных чисел и завершается числом 0. Всего вводится не более 10000 чисел (не считая завершающего числа 0). Определите, сколько элементов этой последовательности равны ее наибольшему элементу.

Оценочный лист к типовому заданию (модельный ответ):

B1. Программа вернет 5. Программа возвращает введенное число, игнорируя остальные символы. Засчитывается любое альтернативное работоспособное решение. Задача оценивается в 10 баллов

B2. Оценивается работоспособность программы и скорость работы алгоритма. Задача оценивается в 10 баллов

Методические материалы, определяющие процедуру и критерии оценивания сформированности компетенций при проведении промежуточной аттестации

Код контролируемой компетенции (индикаторы)	Наименование оценочного средства	Максимальное количество баллов	Всего баллов	Уровень освоения компетенции (в баллах)		
				Пороговый (56-70%)	Продвинутый (71-85%)	Высокий (86-100%)
ПК-1.1	Задание А 1-5	40	40	20-24	28-32	36-40
ПК-1.2	Задание В	20	20	10-14	15-16	17-20

Полученное число баллов (30-60) выставляется в графу «Промежуточная аттестация» балльно-рейтинговой карты дисциплины.

Вариант 2

A1. Оператор вывода cout может печатать несколько значений или переменных в одной команде, используя следующий синтаксис:

- cout << "Привет", name, "n";
- cout << "Привет" + name + "n";
- cout << ("Привет" & name & "n");
- cout << "Привет" << name << "n";

A2. Какое значение будет напечатано, в результате выполнения программы?

```
#include
int main()
{
    int x = 3;
    switch(x)
    {
        case 0:
            int x = 1;
            std::cout << x << std::endl;
            break;
        case 3:
            std::cout << x << std::endl;
            break;
        default:
            x = 2;
            std::cout << x << std::endl;
    }
    return 0;
}
```

```

}

```

- a) 3
- b) 2
- c) ничего не напечатается, программа вообще не будет работать
- d) 1
- e) 0

A3. Тело оператора выбора if, будет выполняться, если его условие:

- a) ложно (false)
- b) истинно (true)
- c) выполнение не зависит от условия

A4. Укажите блок кода, в котором переменная y доступна.

```

int main(int argc, char** argv)
{
    if ( argc > 10 )
    {
    }
    else if (int y = argc - 1 )
    {
    }
    else
    {
    }
    return 0;
}

```

- a) строки 8 -11
- b) строки 8 -17
- c) строки 8 -15
- d) строки 4 -17
- e) строки 4 -15

A5. Что появится на экране, после выполнения этого фрагмента кода?

```

int a = 1, b =2;
if (a == b);
cout << a << " = " << b << endl;

```

- a) синтаксическая ошибка
- b) 1 = 2
- c) a = b
- d) вывод на экран не выполнится

A6. Результат выполнения следующего фрагмента кода: !((1 || 0) && 0)

- a) результат не может быть заранее определен
- b) 1
- c) 0
- d) 4

A7. Какое из следующих значений эквивалентно зарезервированному слову true?

- a) 0.1
- b) 1
- c) -1
- d) 66
- e) Все варианты ответов

A8. Это значение 5.9875e17 может быть сохранено в переменной, типа

- a) Bool
- b) Float
- c) Int
- d) Long
- e) Short

A9. Вывод данных в C++

- a) cout << <переменная>, <"< строка выводится на экран">, <выражение>, endl;
- b) cout << <переменная>, <"< строка выводится на экран">, <выражение>, endl;

- c) `cout << <переменная> << "< строка выводится на экран>" << <выражение> << endl;`

A10. В каком случае лучше всего использовать приведение типов данных?

- a) во всех выше указанных случаях
- b) чтобы разрешить программе использовать только целые числа
- c) чтобы изменить тип возвращаемого значения функции
- d) при делении двух целых чисел, для того, чтобы вернуть результат с плавающей точкой

A11. Какой тип данных имеет переменная ARGV?

- a) это не переменная
- b) `char **`
- c) `char *`
- d) `int`

A12. Что будет напечатано на экране, после выполнения этого кода?

```
#include
int foo(int y);
int foo(int x)
{
    return x+1;
}
int main(int argc, char** argv)
{
    int x = 3;
    int y = 6;
    std::cout << foo(x) << std::endl;
    return 0;
}
```

- a) 4
- b) ошибка компиляции
- c) 3
- d) 9

A13. Какая строка содержит зарезервированные слова языка программирования C++?

- a) `sizeof, const, typedef, static, voided, enum, struct, union`
- b) `char, int, float, doubled, short, long, unsigned, signed`
- c) `if, else, for, while do, switch, continue, break`
- d) `defaulted, goto, return, extern, private, public, protected`

A14. Что будет напечатано, после выполнения этого кода: `cout << (5 << 3);` ?

- a) 35
- b) 53
- c) 40
- d) 56

A15. Укажите неправильно записанную операцию отношения

- a) `>=`
- b) `<=`
- c) `=!`
- d) все операторы записаны правильно

A16. Результат выполнения следующего фрагмента кода: `cout << 22 / 5 * 3;`

- a) 12
- b) 13.2
- c) 1
- d) 1.47
- e) Другое

A17. В каком случае можно не использовать фигурные скобки в операторе выбора `if`?

- a) если в теле оператора `if` всего один оператор
- b) если в теле оператора `if` два и более операторов
- c) нет правильного ответа
- d) если в теле оператора `if` нет ни одного оператора

A18. Ввод данных в C++

- a) cin » <выражение1> » <выражение2>...;
- b) cin » <выражение1>,<выражение2>,...;
- c) cin » <выражение1> » <выражение2> » endl »...;

A19. Какое ключевое слово указывает, что целая переменная не может принимать отрицательные значения?

- a) Unsigned
- b) Positive
- c) нет такого зарезервированного слова
- d) long
- e) другое

A20. Преобразование целочисленной переменной value в ASCII эквивалент

- a) cout << value
- b) atoi(value)
- c) (char) value
- d) char (value)

Оценочный лист к типовому заданию А

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
d	c	b	c	b	b	e	b	c	d
A11	A12	A13	A14	A15	A16	A17	A18	A19	A20
b	b	c	c	c	b	a	a	a	c,d

Каждый верный ответ оценивается в 2 балла.

Задание В

B1. Оцените следующий алгоритм. Что вернет программа при вводе числа 105. Опишите, что делает программа в общем случае? Предложите альтернативный вариант решения.

```
void prime_divisors(int n)
{
    int t,k;
    cout<<n<<"=";
    if (n<=1)
    {
        cout<<" "<<n;
        return ;
    }
    t=sqrt(n);
    k=2;
    while (k<=t)
    {
        if (!(n%k))
        {
            n/=k;
            cout<<" "<<k;
            while (!n%k)
            {
                n/=k;
                cout<<" "<<k;
            }
            t=sqrt(n);
        }
        k++;
    }
    if (n>1)
        cout<<" "<<n<<endl;
}
```

B2. На электронную почту Вам пришло письмо, подписанное аббревиатурой (первыми буквами фамилии, имени и отчества (далее - ФИО) отправителя). Аббревиатура оказалась Вам незнакома. У Вас есть список всех предполагаемых отправителей, взятый из ранее полученных писем, среди которых различных людей с такой аббревиатурой не больше 10.

Вам предлагается написать эффективную, в том числе по используемой памяти, программу, которая определит всех вероятных адресатов – людей, ФИО которых можно сократить до нужной аббревиатуры. ФИО следует

выдать в порядке убывания частоты их встречаемости в списке, если несколько человек встречаются с одинаковой частотой, то вывести их в алфавитном порядке.

Оценочный лист к типовому заданию (модельный ответ):

В1. Программа вернет $3 \cdot 5 \cdot 7$. Программа раскалывает введенное число на простые сомножители. Засчитывается любое альтернативное работоспособное решение. Задача оценивается в 10 баллов

В2. Оценивается работоспособность программы и скорость работы алгоритма. Задача оценивается в 10 баллов

Методические материалы, определяющие процедуру и критерии оценивания сформированности компетенций при проведении промежуточной аттестации

Код контролируемой компетенции (индикаторы)	Наименование оценочного средства	Максимальное количество баллов	Всего баллов	Уровень освоения компетенции (в баллах)		
				Пороговый (56-70%)	Продвинутый (71-85%)	Высокий (86-100%)
ПК-1.1	Задание А 1-20	40	40	20-24	28-32	36-40
ПК-1.2	Задание В	20	20	10-14	15-16	17-20

Полученное число баллов (30-60) выставляется в графу «Промежуточная аттестация» балльно-рейтинговой карты дисциплины.

Проверяемая компетенция:

ПК-1. Способен осуществлять педагогическую деятельность по реализации образовательного процесса по предмету

Проверяемый индикатор достижения компетенции:

ПК-1.1. Умеет реализовывать образовательную программу по предмету с учетом специфики содержания, методов и инструментов соответствующей области научного знания

Проверяемые результаты обучения:

Знает: современные методы процедурного, объектно-ориентированного, функционального программирования; основные конструкции языков программирования; основные типы данных и операторы.

Умеет: записывать разрабатываемые алгоритмы на языках программирования высокого уровня; комментировать синтаксические и семантические ошибки, возникающие при некорректном выполнении программы; отлаживать и тестировать задачи; составлять систему тестов для автоматизированной проверки корректности программы; подбирать оптимальное программное обеспечение для разработки, тестирования и отладки программ.

Проверяемый индикатор достижения компетенции:

ПК-1.2. Реализует образовательную программу по предмету с использованием технологий профессиональной деятельности

Проверяемые результаты обучения:

Умеет: использовать знания по программированию при реализации образовательной программы по информатике.

Комплект оценочных средств для проведения промежуточной аттестации в 4 семестре

Раздел 2. Объектно-ориентированное программирование

1. Какие из следующих утверждений об исключениях в Python верны? (Отметьте все варианты)

1. Исключения-наследники Exception являются проверяемыми.
2. Все исключения в Python являются непроверяемыми (не требуется объявлять в сигнатуре).
3. Исключения-наследники ArithmeticError относятся к ошибкам времени выполнения.
4. Исключения можно перехватывать с помощью блока except.
5. Исключения можно перехватывать с помощью блока catch.

2. Какие варианты корректно заполняют пропуски в этом коде? (Выберите все подходящие)

```
python
```

```
def divide(a, b):
    if b == 0:
        _____ ZeroDivisionError("Деление на ноль")

    _____
    return a / b
```

1. Вставить raise
2. Вставить throw
3. Вставить return
4. Вставить except
5. Вставить raise new

3. Когда использование блока finally в конструкции try-except-finally является обязательным?

1. Никогда, блок finally всегда опционален.
2. Когда код должен выполнить очистку ресурсов независимо от возникновения исключения.
3. Когда в конструкции try нет блоков except.
4. Когда в конструкции try есть ровно один блок except.
5. Когда в конструкции try есть два или более блоков except.

4. Какое исключение возникнет при выполнении следующего кода?

```
python
```

```
def get_value(data, index):
```

```
return data[index]
```

```
my_list = [1, 2, 3]
```

```
result = get_value(my_list, 5)
```

1. IndexError
2. TypeError
3. ValueError
4. KeyError
5. AttributeError

5. Какие утверждения об обработке исключений в Python верны? (Отметьте все варианты)

1. Можно перехватывать несколько типов исключений в одном блоке except.
2. Блок else выполняется только если исключение не возникло.
3. Блок finally выполняется всегда, даже если исключение не было перехвачено.
4. Ключевое слово throws используется для объявления исключений в сигнатуре метода.
5. Исключения в Python являются классами и наследуются от BaseException.

6. Тест, который проверяет, что функция деления при запросе 2/0 возвращает ZeroDivisionError, это:

1. Модульный тест (unit-тест)
2. Нагрузочный тест
3. Функциональный тест
4. Интеграционный тест

7. Любой поток выполнения (thread) в Python — это объект?

1. Да
2. Нет

8. Декоратор @deprecated на методе класса означает:

1. что метод устарел и его использование не рекомендуется
2. что в методе есть синтаксическая ошибка
3. что метод будет удален в следующей версии
4. что метод помечен для отладки

9. Все ли классы для чтения данных из файлов наследуются от базового класса IOBase (или его наследников)?

1. Да, все классы для работы с потоками ввода-вывода наследуются от IOBase
2. Нет, некоторые классы являются независимыми
3. Только классы для текстовых файлов
4. Только классы для бинарных файлов

10. Синхронное обращение к базе данных означает:

1. приложение передает в базу callback, который будет вызван базой после выполнения запроса
2. приложение ожидает завершения выполнения запроса и получения результата
3. приложение ждет, пока будет закрыто соединение к базе
4. запрос выполняется в отдельном потоке

11. Метод, декорированный @before, в фреймворке для тестирования (например, pytest) будет запущен:

1. один раз перед стартом всех тестов в сессии
2. один раз перед запуском тестов в конкретном классе
3. перед запуском каждого тестового метода в классе
4. после каждого тестового метода

12. Какие варианты решения проблемы взаимодействия потоков могут быть расширены на взаимодействие процессов (запущенных на разных компьютерах)?

1. держать в каждом потоке ссылки на все остальные потоки
2. менять состояние общего для потоков объекта
3. передавать сообщения через общий ресурс
4. передавать сообщения через систему обмена сообщениями (например, очереди)

13. Сборщик мусора (Garbage Collector) в Python — это:

1. механизм, который автоматически удаляет объекты, на которые нет ссылок
2. часть компилятора, которая расставляет операторы del
3. механизм, который удаляет локальные переменные при выходе из функции
4. механизм, который удаляет объекты, недостижимые из корневых ссылок (используя подсчет ссылок и циклический сборщик)

14. Будет ли сериализовано статическое поле класса при использовании модуля pickle?

1. Да, всегда
2. Нет, статические поля не сериализуются
3. Да, если его отметить специальным атрибутом
4. Только если оно является изменяемым

15. Тест, который проверяет, что два пользователя могут зайти в игру и начать партию в шахматы, это:

1. Нагрузочные тесты
2. Модульные тесты (unit-тесты)
3. Функциональные тесты
4. Тесты производительности

16. Какие утверждения относительно обработки событий в GUI-библиотеках (например, Tkinter) верны?

1. События можно использовать для отслеживания добавления новых виджетов в контейнер
2. Событие нажатия клавиши клавиатуры можно отслеживать через обработчик событий
3. Событие изменения состояния окна сигнализирует об изменении его размера или положения
4. Обработчики событий можно использовать для закрытия окна приложения

17. Для чего нужны классы-обертки в Python (например, int, float, bool)?

1. использование классов-обертки ускоряет работу приложения
2. переменная класса-обертки может принимать значения None
3. классы-обертки предоставляют набор методов для работы с данными
4. все данные в Python являются объектами, поэтому классы-обертки используются по умолчанию

18. Почему объектно-ориентированное программирование пришло на смену процедурному программированию?

1. использование ООП позволяет создавать не зависящие от платформы программы
2. ООП позволяет уменьшить сложность создания ПО за счет инкапсуляции и повторного использования кода
3. программы, написанные с использованием ООП, выполняются на компьютере быстрее
4. ООП позволяет увеличить надежность создаваемого ПО за счет инкапсуляции и полиморфизма

19. Какие утверждения относительно полей (атрибутов) класса в Python верны?

1. Атрибут, объявленный с именем, начинающимся с двух подчеркиваний (__attr), подвергается name mangling
2. Атрибуты класса можно объявлять с модификатором final для запрета переопределения
3. Атрибут экземпляра ссылочного типа имеет значение по умолчанию None
4. Атрибуты не могут иметь модификатор доступа protected (по соглашению используется _attr)

20. Предположим, вы моделируете автомобиль, описывая его свойства в формате Python-класса. Какие из следующих полей нужно объявить атрибутами экземпляра (не класса)?

1. максимально допустимая масса для этого класса автомобилей
2. дата начала выпуска автомобилей
3. идентификационный номер (VIN)
4. дата выпуска автомобиля

21. Какие утверждения относительно буферизованного ввода-вывода в Python (например, `BufferedReader`, `BufferedWriter`) верны?

1. позволяет производить многократную запись небольших блоков данных без обращения к устройству вывода
2. запись в буфер автоматически сбрасывается на устройство вывода при каждом обращении
3. можно явно вызвать сброс буфера методом `flush()`
4. запись с использованием буфера значительно быстрее, чем без него

22. Что можно вставить на место пропуска в этом коде, чтобы определение метода было корректным? (Отметьте все варианты)

```
python
```

```
class Ant:
    _____ def method(self):
```

```
pass
```

1. `@staticmethod`
2. `@classmethod`
3. `@property`
4. `@abstractmethod`
5. `@final`
6. `async`

23. Какие из перечисленных определений методов в Python являются синтаксически корректными? (Отметьте все варианты)

1. `def method1(self): return 42`
2. `def method2(self) -> int: return 42`
3. `def method3(self) -> None: return`
4. `@staticmethod def method4(): return 42`
5. `@classmethod def method5(cls): return cls.__name__`
6. `def method6(self) -> int: return "строка"` (синтаксически корректно, но не по типам)

24. Какие из перечисленных определений методов в Python являются синтаксически корректными? (Отметьте все варианты)

1. `def method_a(self): return`
2. `def method_b(self): return None`
3. `def method_c(self): pass`
4. `def method_d(self): return 9`
5. `def method_e(self): return 9.0`
6. `def method_f(self): return True`
7. `def method_g(self): return "text"`

25. Какие из перечисленных сигнатур функций с переменным числом аргументов в Python являются корректными? (Отметьте все варианты)

1. `def func(self, *args): pass`
2. `def func(self, first, *args): pass`
3. `def func(self, *args, last): pass`

4. `def func(self, **kwargs): pass`
5. `def func(self, *args, **kwargs): pass`
6. `def func(self, first, *args, **kwargs): pass`
7. `def func(self, first, second): pass`

26. Дан следующий метод. Какие вызовы метода возвращают 2? (Отметьте все варианты)

python

```
def how_many(self, b, *b2):
```

```
    return len(b2)
```

1. `how_many()`
2. `how_many(True)`
3. `how_many(True, True)`
4. `how_many(True, True, True)`
5. `how_many(True, [True])`
6. `how_many(True, (True, True))`
7. `how_many(True, True, True, True)`

27. Какие утверждения о модификаторах доступа в Python верны? (Отметьте все варианты)

1. Атрибут с одним подчеркиванием в начале (`_attr`) считается защищенным по соглашению
2. Атрибут с двумя подчеркиваниями в начале (`__attr`) полностью недоступен извне
3. Все атрибуты в Python являются публичными по умолчанию
4. Можно использовать механизм свойств (`@property`) для контроля доступа к атрибутам
5. В Python существуют модификаторы доступа `public`, `private`, `protected`

28. Даны определения классов. Какие строки в коде вызовут ошибку доступа к атрибутам? (Отметьте все варианты)

python

```
# classroom.py
class Classroom:
    def __init__(self, r, t):
        self.__room_number = r    # приватный атрибут
        self._teacher_name = t    # защищенный атрибут
        self.floor = 3            # публичный атрибут

    _global_key = 54321           # защищенный атрибут класса
```

```
# main.py
from classroom import Classroom

room = Classroom(101, "Mrs. Anderson")
print(Classroom._global_key)    # строка 5
print(room.__room_number)       # строка 6
print(room.floor)               # строка 7
```

```
print(room._teacher_name)       # строка 8
```

1. Нет ошибок, код выполняется корректно
2. Строка 5
3. Строка 6
4. Строка 7
5. Строка 8
6. Все строки вызовут ошибку

29. Какие утверждения об инкапсуляции и неизменяемости в Python верны? (Отметьте все варианты)

1. Инкапсуляция предполагает использование защищенных (`_attr`) атрибутов экземпляра
2. Инкапсуляция предполагает использование приватных (`__attr`) атрибутов экземпляра
3. Инкапсуляция позволяет использовать методы-геттеры и сеттеры
4. Неизменяемость предполагает использование защищенных (`_attr`) атрибутов экземпляра
5. Неизменяемость предполагает использование приватных (`__attr`) атрибутов экземпляра
6. Неизменяемость допускает использование сеттеров для изменения состояния

30. Какие методы соответствуют соглашениям Python для свойств (`property`) и их сеттеров? (Отметьте все варианты)

1. `def get_can_swim(self): return self._can_swim`
2. `def can_swim(self): return self._can_swim`
3. `def get_num_wings(self): return self._num_wings`
4. `def num_wings(self): return self._num_wings`
5. `def set_can_swim(self, value): self._can_swim = value`

31. Что выведет (в консоль) следующий код?

python

```
# rope.py
class Rope:
    LENGTH = 5

    @staticmethod
    def swing():
        print("swing ", end="")

# main.py
from rope import Rope

class Chimp:
    @staticmethod
    def main():
        Rope.swing()
        Rope().swing()
        print(Rope.LENGTH)
```

Chimp.main()

Ответ: swing swing 5

32. Какие утверждения корректны по отношению к данному коду? (Отметьте все варианты)

python

```
class Rope:
    @staticmethod
    def swing():
        print("swing ", end="")

    def climb(self):
        print("climb ", end="")

    @staticmethod
    def play():
```

```
Rope.swing()
Rope().climb() # корректный вызов нестатического метода
```

```
@staticmethod
def main():
    rope = Rope()
    rope.play()
    rope2 = None
    # rope2.play() # вызовет ошибку, т.к. None не имеет метода
```

Rope.main()

1. Код выполняется без ошибок
2. В коде есть одна ошибка выполнения
3. Вывод программы: swing climb swing climb
4. Вывод программы: swing swing

33. Какие модификаторы неявно применяются к методам абстрактных базовых классов (ABC) в Python? (Отметьте все варианты)

1. @abstractmethod
2. @staticmethod
3. @classmethod
4. @property
5. @final
6. Методы не имеют неявных модификаторов, всё определяется явно

34. Что выведет (в консоль) следующий код?

python

```
class Mammal:
    def __init__(self, age):
        print("Mammal", end="")

class Platypus(Mammal):
    def __init__(self):
        print("Platypus", end="")

    @staticmethod
    def main():
        Mammal(5)
```

Platypus.main()

1. Platypus
2. Mammal
3. PlatypusMammal
4. MammalPlatypus
5. Ошибка в строке, где определен __init__ в Platypus
6. Ошибка в строке вызова Mammal(5)

35. Что можно вставить на место подчеркивания, чтобы код выполнялся корректно? (Отметьте все варианты)

python

```
class CanHop: pass
```

```
class Frog(CanHop): pass
class BrazilianHornedFrog(Frog): pass
class TurtleFrog(Frog): pass

if __name__ == "__main__":
```

```
    frog = _____()
```

1. Frog
2. TurtleFrog
3. BrazilianHornedFrog
4. CanHop
5. object
6. int

36. Какие утверждения верны относительно следующего кода? (Отметьте все варианты)

```
python
```

```
class Rodent:
    @staticmethod
    def chew():
        print("Rodent is chewing")
        return 1

class Beaver(Rodent):
    @staticmethod
    def chew():
        print("Beaver is chewing on wood")

        return 2
```

1. Код выполняется без ошибок
2. Метод в подклассе переопределяет метод родителя
3. Статические методы не могут быть переопределены
4. Возвращаемые типы должны совпадать
5. Код вызовет ошибку из-за несоответствия декораторов

37. Что может быть только скрыто (hidden) в подклассе, но не переопределено (overridden)? (Отметьте все варианты)

1. Приватные методы с именем, начинающимся с __
2. Защищенные методы с именем, начинающимся с _
3. Публичные методы
4. Статические методы (@staticmethod)
5. Атрибуты класса
6. Приватные атрибуты с именем, начинающимся с __

38. Выберите корректное утверждение о данном коде:

```
python
```

```
from abc import ABC, abstractmethod

class HasExoskeleton(ABC):
    @abstractmethod
    def get_number_of_sections(self):
        pass
```

```
class Insect(HasExoskeleton):
    @abstractmethod
    def get_number_of_legs(self):
        pass
```

```
class Beetle(Insect):
    def get_number_of_legs(self):
```

```
        return 6
```

1. Код выполняется без ошибок
2. Ошибка в строке определения get_number_of_sections
3. Ошибка в строке определения Insect
4. Ошибка в строке определения Beetle (не реализован абстрактный метод)
5. Код выполняется, но вызывает исключение во время выполнения

39. Какие утверждения о полиморфизме в Python верны? (Отметьте все варианты)

1. Переменная может ссылаться на объект любого типа без явного приведения
2. Если функция принимает объект суперкласса, то можно передать объект любого подкласса
3. Функция, принимающая параметр типа object, примет любой объект
4. Все ошибки типов могут быть обнаружены во время компиляции (статическая типизация)
5. Утиная типизация (duck typing) позволяет вызывать методы, даже если объект не наследует нужный

класс

40. Какие утверждения верны как для абстрактных классов, так и для интерфейсов (протоколов) в Python? (Отметьте все варианты)

1. Все методы в них предполагаются абстрактными
2. Оба могут содержать атрибуты класса
3. Оба могут быть расширены через наследование
4. Оба могут содержать методы с реализацией по умолчанию
5. Оба могут содержать статические методы
6. Ни один из них не может быть инстанцирован напрямую
7. Оба наследуются от object

Правильные ответы:

1. 2, 3, 4
2. 1
3. 1
4. 1
5. 1, 2, 3, 5
6. 1
7. 1
8. 1
9. 1
10. 2
11. 3
12. 4
13. 1, 4
14. 2
15. 3
16. 2, 3, 4
17. 2, 3, 4
18. 2, 4
19. 1, 3
20. 3, 4
21. 1, 3, 4
22. 1, 2, 3, 4, 6

- 23. 1, 2, 3, 4, 5, 6
- 24. 1, 2, 3, 4, 5, 6, 7
- 25. 1, 2, 3, 4, 5, 6, 7
- 26. 3, 7
- 27. 1, 3, 4
- 28. 3 (строка 6)
- 29. 2, 3, 5
- 30. 2, 4
- 31. swing swing 5
- 32. 1, 3
- 33. 6
- 34. 2
- 35. 1, 2, 3, 4, 5 (все кроме 6)
- 36. 1, 2
- 37. 1, 4, 5, 6
- 38. 4
- 39. 1, 2, 3, 5
- 40. 2, 3, 5, 6

Оценочный лист к заданию 1.1.

Критерий	Максимальное количество баллов
Вопрос 1-40	1

Задание 1.2.

Содержание задания:

Создайте программу, осуществляющую чтение текстового файла и загрузку всех его слов (разделители – пробелы, знаки препинания удаляются) в соответствующую коллекцию. Организовать подсчет чаще всего встречающегося слова. Интерфейс программы – нативное десктоп-приложение.

В программе должны присутствовать методы для чтения и загрузки слов файла в коллекцию, для преобразования коллекции в структуру «ключ»-«значение», где «значение» – частота встречаемости слова, для поиска чаще всего встречающихся слов. Предусмотреть автоматизированное тестирование средствами JUnit всех указанных методов.

Окно диалога программы должно включать поле ввода для имени файла (с вызовом диалога открытия файла ОС), а также текстовое поле для вывода чаще всего встречающихся слов. Провести ручное тестирование для случаев а) корректного и некорректного имени файла б) пустого файла в) файла, в которых слова встречаются по одному разу.

Оформить результаты эргономического тестирования приложения.

Оценочный лист к заданию 1.2.

Показатель результативности	Индикатор ПК-1.2	Максимальное количество баллов
Приложение создано, отлажено, работает, проходит тесты при корректных входных данных	ПК-1.2	5
Подготовлены тесты JUnit и успешно пройдены	ПК-1.2	5
Выполнено (и пройдено) ручное тестирование	ПК-1.2	5
Дано развернутое методическое объяснение применению концепций и возможностей ООП	ПК-1.2	5

Методические материалы, определяющие процедуру и критерии оценивания сформированности компетенций при проведении промежуточной аттестации

Код контролируемой компетенции (индикаторы)	Наименование оценочного средства	Максимальное количество баллов	Всего баллов	Уровень освоения компетенции (в баллах)		
				Пороговый (56-70%)	Продвинутый (71-85%)	Высокий (86-100%)
ПК-1.1	Задание А 1-40	40	40	20-24	28-32	36-40
ПК-1.2	Задание В	20	20	10-14	15-16	17-20

Полученное число баллов (30-60) выставляется в графу «Промежуточная аттестация» балльно-рейтинговой карты дисциплины.