Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования «Кабардино-Балкарский государственный университет им. Х.М. Бербекова»
(КБГУ)

Институт информатики, электроники и робототехники
Кафедра электроники и цифровых информационных технологий

ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ

Сертификат: 00AF3CD5350461AF78E75D191E45B0C3AC
Владелец: ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ "КАБАРДИНО-БАЛКАРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. Х. М. БЕРБЕКОВА"
Действителен: с 27.03.2024 до 20.06.2025

УТВЕРЖДАЮ
Руководитель ОПОП
О.А. Молоканов
«16» декабря 2024 г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ (ОЦЕНОЧНЫХ МАТЕРИАЛОВ) ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

«Объектно-ориентированное программирование»

Программа специалитета
12.05.01 Электронные и оптико-электронные приборы и системы специального назначения

Специализация
Оптико-электронные информационно-измерительные приборы и системы

Форма обучения
Очная

Квалификация (степень выпускника)
инженер

Нальчик 2024

СОДЕРЖАНИЕ

1.Перечень компетенций с указанием этапов их формирования в процессе освоения образовательной программы, описание показателей, критериев оценивания компетенций на различных этапах их формирования

Карта компетенции

Общепрофессиональные компетенции

ОПК-4. Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения.

Код и наименование индикатора достижения компетенции:

ОПК-4.1. Способен разрабатывать алгоритмы решения задач своей профессиональной деятельности.

ОПК-4.2. Способен разрабатывать программное обеспечение для решения задач своей профессиональной деятельности.

Тип компетенций: общепрофессиональные компетенции выпускника образовательной программы по специальности **12.05.01 «Электронные и оптико-электронные приборы и системы специального назначения»,** специализация **«Оптико-электронные информационно-измерительные приборы и системы»**, уровень ВО – специалитет.

1.1. Этапы формирования компетенций и средства оценивания

Результаты обучения (компетенции)	**Основные показатели оценки результатов обучения**	**Виды оценочных материалов, обеспечивающих формирование компетенций**
ОПК-4. Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения. ***Код и наименование индикатора достижения компетенции:*** ***ОПК-4.1.*** Способен разрабатывать алгоритмы решения задач своей профессиональной деятельности. ***ОПК-4.2.*** Способен разрабатывать программное обеспечение для решения задач своей профессиональной деятельности.	**Знать** основы программирования: языки программирования (Python, Java, C++, и др.), принципы работы компьютера на низком уровне (процессор, память, устройства ввода/вывода).	Выполнение и защита лабораторных работ. Оценочные материалы для практических занятий. Оценочные материалы для коллоквиума. Оценочные материалы для проведения тестирования. Оценочные материалы для промежуточной аттестации.
	Уметь работать с разными инструментами и средами разработки (IDE, системы контроля версий, библиотеки и т.д.)	Выполнение и защита лабораторных работ. Оценочные материалы для практических занятий. Оценочные материалы для коллоквиума. Оценочные материалы для проведения тестирования. Оценочные материалы для промежуточной аттестации.
	Владеть навыками работы с базами данных и написание эффективных запросов.	Выполнение и защита лабораторных работ. Оценочные материалы для практических занятий. Оценочные материалы для коллоквиума. Оценочные материалы для проведения тестирования. Оценочные материалы для промежуточной аттестации.

1.2. Критерии формирования оценок на различных этапах их формирования

Текущий и рубежный контроль

Оценка регулярности, своевременности и качества выполнения обучающимся учебной работы по изучению дисциплины в течение периода изучения дисциплины (сумма – не более 70 баллов). Баллы, характеризующие успеваемость обучающегося по дисциплине, набираются им в течение всего периода обучения за изучение отдельных тем и выполнение отдельных видов работ. Общий балл складывается в результате проведения текущего и рубежного контроля по дисциплине:

Этап (уровень)	**Первый этап (уровень)**	**Второй этап (уровень)**	**Третий этап (уровень)**
Баллы	**36-50 баллов**	**51-60 баллов**	**61-70 баллов**
Характеристика	Полное или частичное посещение аудиторных занятий. Частичное выполнение лабораторных работ. Выполнение контрольных работ, тестовых заданий на оценку «удовлетворительно».	Полное или частичное посещение аудиторных занятий. Полное выполнение лабораторных работ. Выполнение контрольных работ, тестовых заданий на оценки «хорошо».	Полное посещение аудиторных занятий. Полное выполнение лабораторных занятий. Выполнение контрольных работ, тестовых заданий на оценки «отлично».

На первом (начальном) этапе формирования компетенции формируются знания, умения и навыки, составляющие базовую основу компетенции, без которой невозможно ее дальнейшее развитие. Обучающийся воспроизводит термины, факты, методы, понятия, принципы и правила; решает учебные задачи по образцу.

На втором (основном) этапе формирования компетенции приобретается опыт деятельности, когда отдельные компоненты компетенции начинают «работать» в комплексе и происходит выработка индивидуального алгоритма продуктивных действий, направленных на достижение поставленной цели. На этом этапе обучающийся осваивает аналитические действия с предметными знаниями по конкретной дисциплине, способен самостоятельно решать учебные задачи, внося коррективы в алгоритм действий, осуществляя координирование хода работы, переносит знания и умения на новые условия.

Третий (завершающий) этап – это овладение компетенцией. Обучающийся способен использовать знания, умения, навыки при решении задач повышенной сложности и в нестандартных условиях. По результатам этого этапа обучающийся демонстрирует итоговый уровень сформированности компетенции.

Промежуточная аттестация (зачет)

Оценка	**Не зачтено**	**Зачтено**
Баллы	**36-60**	**61-70**
Характеристика	Обучающийся имеет 36-60 баллов по итогам текущего и рубежного контроля. На зачете не выполнил предложенное преподавателем задание. По итогам промежуточного контроля получил 0 баллов.	Обучающийся имеет 36-50 баллов по итогам текущего и рубежного контроля, на зачете полностью выполнил 1/3 и более предложенного преподавателем задания. По итогам промежуточного контроля получил от 11 до 25 баллов. Обучающийся имеет 51-60 баллов по итогам текущего и рубежного контроля, на зачете выполнил одно задание полностью либо частично выполнил 2 из трех заданий. По итогам промежуточного контроля получил от 1 до 10 баллов. Обучающемуся, имеющему 61-70 баллов по итогам текущего и рубежного контроля, выставляется отметка «зачтено» без сдачи зачета.

Промежуточная аттестация (экзамен)

Оценка	Удовлетворительно	Хорошо	Отлично
Баллы	**61 – 80**	**81 – 90**	**91 – 100**
Характеристика	Знает отдельные перспективные задачи в соответствующем научном направлении. Неуверенно докладывает известные результаты в данной предметной области. Готов изложить свои результаты в письменной форме.	Может указать некоторые научные направления, представляющие теоретический и практический интерес. Хорошо представляет известные научные результаты по профилю подготовки. Может устно и письменно изложить свои результаты.	Хорошо ориентируется в современных научных направлениях, соответствующих профильной предметной области. Доказательно и аргументировано представляет собственные и известные научные результаты в данной предметной области. Убедительно и аргументировано излагает свои собственные результаты, как в устной, так и в письменной форме.

2.Методические материалы и типовые контрольные задания, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций в процессе освоения образовательной программы
Перечень оценочных средств

№	Наименование оценочного средства	Краткая характеристика оценочного средства	Представление оценочного средства в фонде
1.	Коллоквиум	Средство контроля усвоения учебного материала темы, раздела или разделов дисциплины.	Вопросы по темам / разделам дисциплины
2.	Контрольная работа	Средство проверки умений применять полученные знания для представления материала по некоторой теме / решения задач определенного типа по некоторому разделу	Вопросы по темам / разделам дисциплины
3.	Лабораторная работа	Средство оценки умения применять полученные теоретические знания в практической ситуации. Задание по работе должно быть направлено на оценивание тех компетенций, которые подлежат освоению в данной дисциплине, и должно содержать четкую инструкцию по выполнению или алгоритм действий.	Перечень лабораторных работ
4.	Тест	Система стандартизированных заданий, позволяющая автоматизировать процедуру измерения уровня знаний и умений обучающегося.	Фонд тестовых заданий

3. Перечень контрольных заданий и иных материалов, необходимых для оценки знаний, умений, навыков и опыта деятельности

3.1. Вопросы для коллоквиумов и контрольных работ
(контролируемые компетенции ОПК-4)

5 семестр

Вопросы для 1 коллоквиума

1. Алфавит языка.
2. Комментарии.
3. Типы данных.
4. Константы-строки, или литералы.

5. Директива препроцессора define.
6. Описание. Модификатор *const.*
7. Операция и выражение присваивания.
8. Арифметические операции.
9. Операции отношения.
10. Логические операции.
11. Побитовые операции.
12. Сдвиги.
13. Операции автоувеличения и автоуменьшения ++ и – .
14. Тернарная, или условная, операция.
15. Операция следования.
16. Приоритеты операций и порядок вычисления

Вопросы для 2 коллоквиума

1. Определение указателей.
2. Указатели и массивы.
3. Адресная арифметика.
4. Символьные массивы и строки.
5. Многомерные массивы.
6. Указатели и многомерные массивы.
7. Операция выделения памяти *new.*
8. Операция освобождения памяти *delete.*
9. Компиляция, компоновка, библиотеки.
10. Виды областей существования имени.

Вопросы для 3 коллоквиума

1. Явные преобразования.
2. Неявные преобразования стандартных базовых типов.
3. Преобразование производных стандартных типов.
4. Определение и вызов функции.
5. Передача аргументов в функцию.
6. Передача многомерных массивов.
7. Указатели на функции.
8. Ссылки.
9. Ссылки в качестве параметров функций.
10. Рекурсивные функции.
11. Аргументы по умолчанию.
12. Перегрузка функций.
13. Шаблоны функций.
14. Объявление классов.
15. Конструкторы. Деструкторы.
16. Статические члены класса.
17. Указатель *this.* Статические функции-члены.
18. Указатели на члены класса. Инициализация данных-членов класса.
19. Конструктор копирования и операция присваивания.
20. Дружественные функции. Конструктор и операция *new* . Вызов деструктора.

6 семестр

Вопросы для 1 коллоквиума

1. Построение производного класса.
2. Защищенные члены класса.
3. Управление уровнем доступа к членам класса.
4. Последовательность вызова конструктора и деструктора при построении производного класса на основе одного базового.
5. Преобразования типов.
6. Раннее и позднее связывание.

7. Виртуальные функции.
8. Абстрактные классы.
9. Основные определения и свойства класса и шаблона.
10. Операции *new* и *delete* при работе с абстрактными типами.
11. Использование *new* при создании динамического объекта абстрактного типа.
12. Операция *delete.*
13. Преобразование типов.

Вопросы для 2 коллоквиума

1. Особенности переопределенных операций
2. Операция = . Операция [ ]. Классы Array и Matrix.
3. Операции над односвязными списками.
4. Двунаправленные и кольцевые списки.
5. Операции над кольцевыми списками.
6. Реализация стека через массив.
7. Реализация стека через динамическую цепочку звеньев.
8. Определение и построение. Таблицы.

Вопросы для 3 коллоквиума

1. Классы потоков.
2. Стандартные потоки.
3. Операции помещения и извлечения из потока.
4. Форматирование потока.
5. Файловый ввод-вывод с использованием потоков.
6. Неформатируемый ввод-вывод.
7. Часто применяемые функции.
8. Файлы с произвольным доступом.
9. Опрос и установка состояния потока.
10. Переопределение операций извлечения и вставки.
11. Переадресация ввода-вывода.

Рекомендации при подготовке к коллоквиуму

- проработать конспекты лекций по вопросам коллоквиума;
- прочитать основную и дополнительную литературу, рекомендованную по изучаемым вопросам;
- ответить на вопросы коллоквиума;
- при затруднениях, проконсультироваться с преподавателем.

3.2. Критерии оценивания

Оценка			
Неудовлетвори-тельно 2 балла	**удовлетворительно 4 балла**	**хорошо 6 баллов**	**отлично 8 баллов**
Студент не знает значительной части вопросов, допускает существенные ошибки в ответах на вопросы	Студент поверхностно знает вопросы коллоквиума, допускает неточности в ответе на вопрос	Студент хорошо знает материал, грамотно и по существу излагает его, допуская некоторые неточности в ответе на вопрос.	Студент в полном объеме знает материал, грамотно и по существу излагает его, не допуская существенных неточностей в ответе на вопрос.

Методические рекомендации по выполнению контрольной работы

При выполнении заданий необходимо внимательно ознакомиться с контентом по вопросам соответствующей темы. Основная цель работы – овладеть навыками исследования изучаемого вопроса.

3.2. Типовые тестовые задания по дисциплине

(контролируемые компетенции ОПК-4)

5 семестр

1-я контрольная точка

I:
S: Типы bool, int, char являются …
- фундаментальными типами
- перечислениями
- структурами
- типами, определяемыми пользователем

I:
S: Типы float, double являются …
- интегральными типами
- типами с плавающей точкой
- классами
- типами, определяемыми пользователем

I:
S: Типы bool, int, char являются …
- интегральными типами
- типами с плавающей точкой
- объединениями
- типами, определяемыми пользователем

I:
S: Классы и структуры являются …
- арифметическими типами
- интегральными типами
- символьными типами
- типами, определяемыми пользователем

I:
S: Типы double, int, char являются …
- типами с плавающей точкой
- символьными типами
- арифметическими типами
- интегральными типами

I:
S: Перечисления и объединения являются …
- арифметическими типами
- интегральными типами
- символьными типами
- типами, определяемыми пользователем

I:
S: Создайте переменную с произвольным именем, являющуюся указателем на константные данные типа с плавающей точкой.

I:
S: Создайте переменную целого типа с произвольным именем.
Создайте переменную, являющуюся константной ссылкой на первую переменную.

I:
S: Создайте переменную с произвольным именем, являющуюся статическим массивом символов (размер произвольный).

I:
S: Создайте переменную логического типа с произвольным именем.

Создайте вторую переменную, хранящую адрес первой переменной.
I:
S: Создайте переменную с произвольным именем и интегральным типом.
Создайте вторую переменную, являющуюся ссылкой на первую переменную.
I:
S: Создайте переменную с произвольным именем, являющуюся статическим двумерным массивом целых чисел (размеры произвольные).
I:
S: Создайте переменную с произвольным именем, являющуюся константным указателем на константные данные целого типа.
I:
S: Создайте переменную логического типа с произвольным именем.
Создайте вторую переменную, являющуюся ссылкой на первую переменную.

2-я контрольная точка

I:
S: Оператор, позволяющий узнать размер типа на используемой платформе:
- typedef
- sizeof
- тернарный оператор
- бинарный оператор

I:
S: Согласно стандарту выполняется следующее соотношение:
- размер(char) = размер(short) = размер(int) = размер(long)
- размер(char) ≤ размер(short) ≤ размер(int) ≤ размер(long)
- размер(char) ≥ размер(short) ≥ размер(int) ≥ размер(long)

I:
S: Согласно стандарту выполняется следующее соотношение:
- размер(float) = размер(double) = размер(long double)
- размер(float) ≤ размер(double) ≤ размер(long double)
- размер(float) ≥ размер(double) ≥ размер(long double)

I:
S: Все размеры в C++ кратны размеру типа …
- int
- char
- float
- short

I:
S: Объявите функцию без параметров, возвращающую указатель на данные целого типа. Имя функции произвольное.
I:
S: Объявите функцию с одним аргументом типа string, передаваемым по ссылке. Возвращаемый тип – логический. Имя функции произвольное.
I:
S: Объявите не возвращающую ничего функцию с двумя аргументами целого типа, передаваемыми по значению. Имя функции произвольное.
I:
S: Объявите функцию без параметров, возвращающую константную ссылку на значение типа с плавающей точкой. Имя функции произвольное.
I:
S: Объявите функцию с одним аргументом типа double, передаваемым через указатель. Возвращаемый тип – целое число. Имя функции произвольное.
I:
S: Объявите не возвращающую ничего функцию с двумя аргументами целого типа, передаваемыми по константной ссылке. Имя функции произвольное.
I:
S: Какой результат будет выведен в консоль при выполнении функции main?

```
void main() {
  int i=10;
  for (i=i+1; i<12; ++i)
    cout << i;
}
```
- ничего
- 10
- 11
- 1011

I:
S: Какой результат будет выведен в консоль при выполнении функции main?
```
void main() {
  int i = 10;
  if (i = 11)
    cout << i;
}
```
- ничего
- 10
- 11
- 0

I:
S: Какой результат будет выведен в консоль при выполнении функции main?
```
void main() {
  int i=10;
  for (i=i+1; i<11; ++i)
    cout << i;
}
```
- ничего
- 10
- 11
- 1011

I:
S: Какой результат будет выведен в консоль при выполнении функции main?
```
void main() {
  int i=10;
  if (i += 1)
    cout << i;
}
```
- ничего
- 10
- 11
- 0

I:
S: Какой результат будет выведен в консоль при выполнении функции main?
```
void main() {
  int i=11;
  for (i=i–1; i<12; ++i)
    cout << i;
}
```
- ничего
- 10
- 11
- 1011

3-я контрольная точка

I:
S: Какой результат будет выведен в консоль при выполнении функции main?

```
void calc(int x)
{ x = x+5; }

void main() {
  int x=5;
  calc(x);
  cout << x;
}
```
- точно определить невозможно
- 0
- 5
- 10

I:
S: Какой результат будет выведен в консоль при выполнении функции main?
```
void calc(int* p)
{ *p –= 5; }

void main() {
  int x=10;
  calc(&x);
  cout << x–5;
}
```
- точно определить невозможно
- 0
- 5
- 10

I:
S: Какой результат будет выведен в консоль при выполнении функции main?
```
int calc(int& x)
{ return x+5; }

void main() {
  int x=0;
  calc(x);
  cout << calc(x);
}
```
- точно определить невозможно
- 0
- 5
- 10

I:
S: Какой результат будет выведен в консоль при выполнении функции main?
```
int calc(int x)
{ return x–5; }

void main() {
  int x=10;
  x = calc(5);
  cout << x+5;
}
```
- точно определить невозможно
- 0
- 5
- 10

I:
S: Какой результат будет выведен в консоль при выполнении функции main?
```
void calc(int* p)
```

```
{ p = 0; }

void main() {
  int x=10;
  calc(&x);
  cout << x;
}
```
- точно определить невозможно
- 0
- 5
- 10

I:
S: Чему будет равна переменная X?
```
int X(6);
X –= X + 3 * 2;
```
- -6
- 18
- 12
- 6

I:
S: Чему будет равна переменная X?
```
double Y(5.2);
int X = (Y > 5) ? 6 : 5;
```
- 5
- 5.2
- 6
- 6.2

I:
S: Чему будет равна переменная X?
```
int Y(5);
int X = Y – 3 * 2;
```
- 4
- 0
- -1
- 1

I:
S: Чему будет равна переменная X?
```
bool Y(true);
int X = (!Y) ? 3+Y : 5+Y;
```
- 3
- 4
- 5
- 6

I:
S: Чему будет равна переменная X?
```
double X(5.5);
X += 5.5–1;
```
- 5.5
- 11
- 10
- 4.5

6 семестр
1-я контрольная точка

I:
S: В приведенном фрагменте кода происходит …
```
string *pt = new string;
```

delete pt;
- выделение и освобождение памяти под объект типа string
- выделение и некорректное освобождение памяти под объект типа string
- выделение и освобождение памяти под массив объектов типа string
- выделение и некорректное освобождение памяти под массив объектов типа string

I:
S: В приведенном фрагменте кода происходит …
double *Ptr = new double;
delete[] Ptr;
- выделение и освобождение памяти под значение типа double
- выделение и некорректное освобождение памяти под значение типа double
- выделение и освобождение памяти под массив значений типа double
- выделение и некорректное освобождение памяти под массив значений типа double

I:
S: В приведенном фрагменте кода происходит …
string *pt = new string[7];
delete[] pt;
- выделение и освобождение памяти под объект типа string
- выделение и некорректное освобождение памяти под объект типа string
- выделение и освобождение памяти под массив объектов типа string
- выделение и некорректное освобождение памяти под массив объектов типа string

I:
S: В приведенном фрагменте кода происходит …
double *Ptr = new double[10];
delete Ptr;
- выделение и освобождение памяти под значение типа double
- выделение и некорректное освобождение памяти под значение типа double
- выделение и освобождение памяти под массив значений типа double
- выделение и некорректное освобождение памяти под массив значений типа double

I:
S: Метод класса, который вызывается автоматически при создании объекта, называется …
- дружественным
- конструктором
- константным
- деструктором

I:
S: Метод класса, который вызывается автоматически при уничтожении объекта, называется
- дружественным
- конструктором
- константным
- деструктором

I:
S: Конструктор класса может быть …
- константным методом
- виртуальным методом
- и константным, и виртуальным
- никаким из перечисленных

I:
S: Деструктор класса может быть …
- константным методом
- виртуальным методом
- и константным, и виртуальным
- никаким из перечисленных

I:
S: Сколько аргументов может принимать деструктор класса?
- >= 0
- всегда 1

- > 0
- всегда 0

I:
S: Что является возвращаемым типом конструктора класса?
- тип первого поля класса
- тип, указанный при объявлении конструктора
- сам класс
- конструктор ничего не возвращает

I:
S: Сколько аргументов может принимать конструктор класса?
- >= 0
- всегда 1
- всегда 0

I:
S: Что является возвращаемым типом деструктора класса?
- тип первого поля класса
- тип, указанный при объявлении деструктора
- сам класс
- деструктор ничего не возвращает

2-я контрольная точка

I:
S: Какой оператор используется для доступа к открытому содержимому класса при работе с объектом класса?
- .
- ->
- *
- никакой из перечисленных

I:
S: Какой оператор используется для доступа к закрытому содержимому класса при работе с указателем на объект класса?
- .
- ->
- *
- никакой из перечисленных

I:
S: Какой оператор используется для доступа к закрытому содержимому класса при работе с объектом класса?
- .
- ->
- *
- никакой из перечисленных

I:
S: Какой оператор используется для доступа к открытому содержимому класса при работе с указателем на объект класса?
- .
- ->
- *
- никакой из перечисленных

I:
S: Защищенные поля и методы класса находятся после спецификатора …
- public
- friend
- private
- protected

I:
S: Спецификатор private разрешает доступ к содержимому класса …

- только из методов этого класса
- из методов этого класса и методов его наследников
- из любого места программы

I:
S: По умолчанию в классе используется спецификатор доступа …
- public
- protected
- private
- спецификатор доступа не определён

I:
S: Спецификатор public разрешает доступ к содержимому класса …
- только из методов этого класса
- из методов этого класса и методов его наследников
- из любого места программы

I:
S: Закрытые поля и методы класса находятся после спецификатора …
- public
- friend
- private
- protected

I:
S: Спецификатор protected разрешает доступ к содержимому класса …
- только из методов этого класса
- из методов этого класса и методов его наследников
- из любого места программы

I:
S: Открытые поля и методы класса находятся после спецификатора …
- public
- friend
- private
- protected

I:
S: Как будет выглядеть объявление конструктора без параметров для приведенного класса?
```
class Car {
  string name;
public:
  string getName() const;
};
```
- void Car();
- Constructor();
- Car(string s);
- Car();
- string Car();

I:
S: Как будет выглядеть объявление деструктора для приведенного класса?
```
class Cat {
  string name;
public:
  string getName() const;
};
```
- void Cat();
- string ~Cat();
- Destructor();
- ~Cat();
- ~Cat(string s);

I:
S: Как будет выглядеть объявление конструктора без параметров для приведенного класса?

```
class Human {
  string name;
public:
  string getName() const;
};
```
- Constructor();
- Constructor(string s);
- ~Human();
- Human();
- void Human();

I:
S: Как будет выглядеть объявление деструктора для приведенного класса?
```
class House {
  string name;
public:
  string getName() const;
};
```
- Destructor();
- ~House();
- ~Destructor();
- void ~House();
- string ~House();

I:
S: Как будет выглядеть объявление конструктора с параметрами для приведенного класса?
```
class Base {
  string name;
public:
  string getName() const;
};
```
- Base();
- ~Base();
- Base(string s);
- ~Base(string s);
- void Base();

I:
S: Как будет выглядеть объявление конструктора без параметров для приведенного класса?
```
class Boss {
  string name;
public:
  string getName() const;
};
```
- Boss();
- ~Boss();
- Boss(string s);
- ~Boss(string s);
- void Boss();

I:
S: Дружественным функциям разрешен доступ …
- ко всему содержимому класса
- ко всему содержимому класса и содержимому классов-наследников
- к открытому и защищенному содержимому класса
- к открытому содержимому класса
- все перечисленные варианты не верны

I:
S: Вложенному (объявленному в другом классе) классу разрешен доступ …
- ко всему содержимому основного класса
- ко всему содержимому основного класса и содержимому его наследников

- к открытому и защищенному содержимому основного класса
- к открытому содержимому основного класса
- все перечисленные варианты не верны

I:
S: Методы класса могут обращаться …
- ко всему содержимому класса
- к открытому и защищенному содержимому класса
- к открытому содержимому класса
- все перечисленные варианты не верны

I:
S: Методы класса-наследника могут обращаться …
- ко всему содержимому базового класса
- к открытому и защищенному содержимому базового класса
- к открытому содержимому базового класса
- все перечисленные варианты не верны

3-я контрольная точка

I:
S: Внутри константных методов класса можно …
- изменять значения полей
- получать значения полей
- изменять и получать значения полей
- ничего из перечисленного

I:
S: Внутри константных методов класса можно …
- вызывать константные методы класса
- вызывать неконстантные методы класса
- вызывать константные и неконстантные методы класса
- ничего из перечисленного

I:
S: Внутри константных методов класса можно …
- получать значения полей и вызывать неконстантные методы класса
- изменять значения полей и вызывать любые методы класса
- получать значения полей и вызывать константные методы класса
- ничего из перечисленного

I:
S: В функции func происходит ...
```
class Student {
  string Name;
public:
  void setName(string s)
  { Name = s; }
  string getName() const
  { return Name; }
};
void func(Student &A) {
  A.setName("Ivanov");
}
```
- изменение состояния объекта A
- ошибка при компиляции
- получение состояния объекта A
- точно определить невозможно

I:
S: В функции func происходит ...
```
class Calc {
  int val;
public:
```

```
  void setVal(string s);
  int getResult();
};
void func(const Calc& A) {
  int r = A.getResult();
}
```
- изменение состояния объекта A
- ошибка при компиляции
- получение состояния объекта A
- точно определить невозможно

I:
S: В функции func происходит ...
```
class Student {
  string Name;
public:
  void setName(string s)
  { Name = s; }
  string getName() const
  { return Name; }
};
void func(Student &A) {
  string s = A.getName();
}
```
- изменение состояния объекта A
- ошибка при компиляции
- получение состояния объекта A
- точно определить невозможно

I:
S: В функции func происходит ...
```
class Calc {
  int val;
public:
  void setVal(string s);
  int getResult();
};
void func(Calc &A) {
  int r = A.getResult();
}
```
- изменение состояния объекта A
- получение состояния объекта A
- точно определить невозможно

I:
S: В функции func происходит ...
```
class Student {
  string Name;
public:
  void setName(string s)
  { Name = s; }
  string getName() const
  { return Name; }
};
void func(const Student &A) {
  A.setName("Ivanov");
}
```
- изменение состояния объекта A
- ошибка при компиляции
- получение состояния объекта A

- точно определить невозможно
I:
S: В функции func состояние объекта A ...
```
class Calc {
  int val;
public:
  void setVal(string s);
  int getResult() const;
};
void func(Calc &A) {
   int r = A.getResult();
}
```
- изменяется
- не изменяется
- точно определить невозможно
I:
S: В какой строке приведенного фрагмента кода вызывается конструктор копии класса Student?
```
1   Student A;
2   Student B("Ivanov");
3   Student C(B);
4   A = Student("Petrov");
```
- 1
- 2
- 3
- 4
I:
S: Как будет выглядеть объявление конструктора копии для класса Comp?
- Comp(const Comp& x);
- Copy(const Comp& x);
- Comp& Comp();
- Comp Comp(const Comp& x);
I:
S: Какая строка будет использоваться вместо многоточия для защиты от самоприсваивания?
```
Calc& Calc::operator=(const Calc& a) {
   …
  val = a.val;
  return *this;
}
```
- if (this == &a)
- if (this != &a)
- if (this == a)
- if (this != a)
I:
S: В какой строке приведенного фрагмента кода вызывается оператор присваивания класса Student?
```
1   Student X("Ivanov");
2   string s = X.getName();
3   Student Y(X);
4   Y = Student("Petrov");
```
- 1
- 2
- 3
- 4
I:
S: Какая строка будет использоваться вместо многоточия для защиты от самоприсваивания?
```
Cat& Cat::operator=(const Cat& c) {
   …
  name = c.name;
```

```
  return *this;
}
```
- if (*this != &c)
- if (*this == &c)
- if (this != &c)
- if (this == &c)

I:

S: В какой строке приведенного фрагмента кода вызывается конструктор копии класса Car?

```
  void func1(Car z) {  }
  void func2(Car &z) {  }
  …
1   Car A("Car");
2   Car B;
3   func1(A);
4   B = A;
5   func2(B);
```
- 1
- 2
- 3
- 4
- 5

Методические рекомендации

Полный банк тестовых заданий по дисциплине представлен в системе онлайно-обучения на базе программного обеспечения Moodle со встроенной подсистемой тестирования КБГУ (https://open.kbsu.ru). Обучающийся, чтобы пройти тестирование, входит в систему open.kbsu.ru под своим личным логином и паролем, выбирает нужную дисциплину и проходит тестирование.

Критерии формирования оценок по тестовым заданиям:

5 баллов – получают обучающиеся с правильным количеством ответов на тестовые вопросы. Выполнено 100 % предложенных тестовых вопросов;

4 балла – получают обучающиеся с правильным количеством ответов на тестовые вопросы – 80 –99 % от общего объема заданных тестовых вопросов;

3 балла – получают обучающиеся с правильным количеством ответов на тестовые вопросы – 50 –79% от общего объема заданных тестовых вопросов;

2 балла – получают обучающиеся с правильным количеством ответов на тестовые вопросы – менее 26-49 % от общего объема заданных тестовых вопросов.

1 балл – получают обучающиеся с правильным количеством ответов на тестовые вопросы – менее 11-25 % от общего объема заданных тестовых вопросов.

0 баллов – получают обучающиеся с правильным количеством ответов на тестовые вопросы – менее 11 % от общего объема заданных тестовых вопросов.

3.3. Перечень лабораторных работ

(контролируемые компетенции ОПК-4)

5 семестр

№ п/п	Наименование лабораторных работ
1.	Программы с линейным алгоритмом
2.	Целочисленные типы. Использование функций
3.	Работа с данными
4.	Запись числа с плавающей точкой
5.	Работа со строками
6.	Строчно-ориентированный ввод с помощью getline() или get(). Смешивание строкового и числового ввода.
7.	Массивы в С++. Объявление, инициализация, использование
8.	Работа с одномерным массивом.
9.	Работа с двумерным массивом.
10.	Введение в класс string
11.	Указатели. Инициализация указателей. Выделение памяти с помощью операции new
12.	Введение в циклы for
13.	Цикл while. Цикл do while
14.	Использование функций.

6 семестр

№ п/п	Наименование лабораторных работ
1.	Указатели
2.	Подпрограммы. Указатели.
3.	Подпрограммы. Функции
4.	Рекурсивные функции
5.	Передача параметра по ссылке
6.	Динамическое выделение памяти
7.	Особенности мультифайлового программирования
8.	Массивы. Связь указателей и массивов.
9.	Определение максимального элемента и его положения в массиве.
10.	Классы и объекты.
11.	Конструкторы и деструкторы
12.	Инкапсуляция, наследование, полиморфизм
13.	Работа с текстом, файлами, строками
14.	Создание интерфейса пользователя в текстовом режиме
15.	Интерполирование алгебраическими многочленами

Критерии формирования оценок по лабораторным работам:

7 баллов - ставится за практические работы, выполненные полностью без ошибок и недочетов; обучающийся демонстрирует знание теоретического и практического материала по теме лабораторной работы;

6 баллов – ставится за практические работы, выполненные полностью, но при наличии в ней не более одной негрубой ошибки и одного недочета, не более трех недочетов. Обучающийся демонстрирует знание теоретического и практического материала по теме лабораторной работы, допуская незначительные неточности;

5 баллов – ставится за практические работы, если студент правильно выполнил не менее 2/3 всех работ или допустил не более одной грубой ошибки и двух недочетов, не более одной грубой и одной негрубой ошибки, не более трех негрубых ошибок, одной негрубой.

менее 4 баллов – ставится за практические работы, если число ошибок и недочетов превысило норму для оценки 3 или правильно выполнено менее 2/3 всех работ.

3.4. Оценочные материалы для промежуточной аттестации

Вопросы к зачету

(контролируемые компетенции ОПК-4)

5 семестр

1) Основные понятия и сферы применения объектно-ориентированного программирования. Преимущества объектно-ориентированного подхода по отношению к процедурному подходу к разработке программ.
2) Предпосылки и история создания языка C++. Достоинства и недостатки языка C++. Сферы применения языка C++.
3) Компиляторы языка C++. Средства разработки программ на языке C++.
4) Классификация типов данных в языке C++. Целые типы данных. Символьные типы данных. Типы для представления чисел с плавающей точкой. Логический тип данных. Тип void.
5) Оператор sizeof(). Гарантии стандарта по соотношению размеров встроенных типов. Получение информации о встроенных типах данных для конкретной платформы.
6) Классификация типов данных в языке C++. Литералы встроенных типов.
7) Структура и особенности объявления имен в языке C++. Имена (идентификаторы). Область видимости. Вложенные области видимости.
8) Пространства имен. Создание псевдонимов для типов данных, ключевое слово typedef.
9) Инициализация сущностей в языке C++. Имена (идентификаторы). Область видимости.
10) Правила инициализации переменных в разных областях видимости при отсутствии инициализатора.
11) Стандартные операции в языке C++. Арифметические операции. Преобразование типов в арифметических выражениях. Приоритет операций.
12) Логические операции. Побитовые логические операции. Операции присваивания. Операция запятая. Приоритет операций.
13) Управляющие конструкции в языке C++. Оператор ветвления if. Тернарная операция. Оператор выбора вариантов switch.
14) Операторы циклов while и do-while. Оператор цикла for. Операторы прерывания циклов break и continue.
15) Определение и основы работы с указателями и массивами в языке C++. Многомерные массивы. Связь между указателями и массивами, индексация с помощью указателей.
16) Константы. Константные указатели и указатели на константу. Ссылки. Константные ссылки.
17) Перечисления. Структуры. Битовые поля. Объединения. Опережающие объявления. Эквивалентность типов.
18) Объявление и определение функции в языке C++. Передача параметров в функцию по значению. Возврат результатов работы функции.
19) Передача параметров по ссылке и константной ссылке. Передача параметров в функцию через указатель. Возврат результатов работы функции.
20) Передача массивов в качестве параметров функции. Многомерные массивы в качестве аргументов функции.
21) Использование внутри функции переменной из охватывающей области видимости. Статические переменные в функциях. Встраиваемые (inline) функции. Перегрузка функций.
22) Перегрузка функций. Аргументы по умолчанию. Функции с переменным числом аргументов. Указатели на функции.
23) Общие сведения о приведении типов. Приведение типов в стиле языка C. Приведение типов в функциональном стиле. Операторы приведения типов в языке C++.
24) Структура памяти программы. Модель памяти с точки зрения программиста на языке C++. Создание динамических объектов (размещение объектов в куче).
25) Динамические массивы. Многомерные динамические массивы.
26) Организация многофайловых программ в языке C++. Спецификатор extern. Директива #include. Защита от множественного включения заголовочных файлов.
27) Понятие о препроцессоре, директивы препроцессора. Директива #define, макросы. Макросы с параметрами. Директивы условной компиляции.
28) Понятие класса и экземпляра класса (объекта). Поля класса.
29) Объявление класса. Спецификаторы доступа к полям и методам.

30) Определение методов класса внутри и снаружи объявления класса.

Целью промежуточных аттестаций по дисциплине является оценка качества освоения дисциплины обучающимися.

Промежуточная аттестация предназначена для объективного подтверждения и оценивания достигнутых результатов обучения после завершения изучения дисциплины. Осуществляется в конце семестра и представляет собой итоговую оценку знаний по дисциплине «Объектно-ориентированное программирование» в виде проведения зачета.

Промежуточная аттестация может проводиться в устной или письменной форме.

Методические рекомендации по подготовке и процедуре осуществления контроля выполнения

Подготовка к промежуточной аттестации заключается в изучении и тщательной проработке обучающимся учебного материала дисциплины с учетом рекомендованного преподавателем учебно-методического обеспечения. Для обеспечения полноты ответа на вопросы и лучшего запоминания рекомендуется составлять план ответа на каждый вопрос.

Критерии оценивания

Шкала оценивания	
Не зачтено (36-60 баллов)	Зачтено (61-70 баллов)
Обучающийся имеет 36-60 баллов по итогам текущего и рубежного контроля. На зачете не выполнил предложенное преподавателем задание. По итогам промежуточного контроля получил 0 баллов	Обучающийся имеет 36-50 баллов по итогам текущего и рубежного контроля, на зачете полностью выполнил одно задание и частично (полностью) второе задание. По итогам промежуточного контроля получил от 11 до 25 баллов. Обучающийся имеет 51-60 баллов по итогам текущего и рубежного контроля, на зачете выполнил одно задание полностью либо частично выполнил оба задания. По итогам промежуточного контроля получил от 1 до 10 баллов. Обучающемуся, имеющему 61-70 баллов по итогам текущего и рубежного контроля, выставляется отметка «зачтено» без сдачи зачета.

Форма билета для зачета
по учебной дисциплине

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования «Кабардино-Балкарский государственный университет им. Х.М. Бербекова» (КБГУ)

Институт информатики, электроники и робототехники
Кафедра электроники и цифровых информационных технологий
Дисциплина – Объектно-ориентированное программирование

БИЛЕТ № 1

1. Определение и основы работы с указателями и массивами в языке C++. Многомерные массивы. Связь между указателями и массивами, индексация с помощью указателей.
2. Использование внутри функции переменной из охватывающей области видимости. Статические переменные в функциях. Встраиваемые (inline) функции. Перегрузка функций.
3. Организация многофайловых программ в языке C++. Спецификатор extern. Директива #include. Защита от множественного включения заголовочных файлов.

Руководитель ОПОП
к.т.н., доцент _____ О.А. Молоканов

Зав. кафедрой электроники
и цифровых информационных технологий,
д.т.н., профессор _____ Р.Ш. Тешев

Вопросы к экзамену

(контролируемые компетенции ОПК-4)

6 семестр

1) Понятие класса и экземпляра класса (объекта). Объявление класса. Спецификаторы доступа к полям и методам. Поля класса.
2) Определение методов класса внутри и снаружи объявления класса. Отличие методов класса от обычных функций, указатель this. Константные методы.
3) Конструкторы. Конструктор по умолчанию. Список инициализации конструктора. Конструктор копии и оператор присваивания. Защита объектов от копирования.
4) Статические поля. Статические константы в классе. Статические методы.
5) Владение ресурсами, конструкторы и деструктор, идиома RAII.
6) Перегрузка операций. Перегрузка операций operator[] и operator(). Операторы приведения типов.
7) Наследование классов. Терминология, применяемая для описания иерархий наследования. Доступ к полям и методам базового класса при наследовании. Спецификаторы доступа при наследовании.
8) Переопределение методов базового класса. Реализация конструкторов и деструкторов при наследовании. Статические элементы при наследовании. Наследование и вложенные классы.
9) Проблема динамической идентификации типов. Виртуальные функции. Статическое и динамическое связывание. Виртуальный деструктор. Чистые виртуальные функции и абстрактные классы.
10) Множественное наследование. Виртуальные базовые классы.
11) Механизм RTTI. Применение операции dynamic_cast. Операция typeid.
12) Шаблоны функций и классов. Инстанцирование и работа с шаблонными объектами. Параметры шаблонов. Эквивалентность типов.
13) Шаблоны функций. Инстанцирование шаблонных функций. Аргументы шаблонных функций.
14) Параметры шаблонов по умолчанию. Полная и частичная специализации шаблонов. Наследование шаблонов.
15) Обработка исключений при помощи кодов ошибок и недостатки данного подхода.
16) Механизм исключений. Генерация исключений. Перехват исключений, блок try-catch. Повторная генерация исключений.
17) Механизм исключений. Передача исключений из вложенного блока. Необработанные исключения. Спецификация исключений.
18) Организация стандартной библиотеки языка C++. Взаимодействие контейнеров, алгоритмов и итераторов.
19) Стандартная библиотека языка C++. Классификация контейнеров. Отличия последовательных и ассоциативных контейнеров.
20) Стандартная библиотека языка C++. Функциональные объекты. Сравнение функциональных объектов и обычных функций.
21) Стандартная библиотека языка C++. Контейнеры стандартной библиотеки. Требования к элементам контейнера. Общие операции контейнеров. Сравнение контейнеров.
22) Динамический массив (vector). Устройство контейнера. Сложность операций. Размер и емкость. Сравнение с массивами в стиле языка C.
23) Двунаправленная очередь. Устройство контейнера. Сложность операций. Управление памятью и целостность итераторов.
24) Список. Устройство контейнера. Сравнение с контейнером vector. Специфические операции со списками.
25) Множество и мультимножество. Внутреннее устройство. Требования к критерию сортировки. Вычислительная сложность операций с множествами, функции поиска.
26) Отображение и мультиотображение. Внутреннее устройство. Сравнение с множествами. Сложность операций. Отображения в качестве ассоциативных массивов.

27) Адаптеры последовательных контейнеров. Стек. Основной интерфейс стека.
28) Адаптеры последовательных контейнеров. Очередь. Основной интерфейс очереди.
29) Адаптеры последовательных контейнеров. Очередь с приоритетом. Интерфейс очереди с приоритетом.
30) Битовые поля. Операции с битовыми полями. Методы класса bitset.
31) Итераторы стандартной библиотеки. Категории итераторов. Итераторы ввода. Итераторы вывода.
32) Итераторы стандартной библиотеки. Категории итераторов. Прямые, двунаправленные итераторы. Итераторы произвольного доступа.
33) Итераторы стандартной библиотеки. Категории итераторов. Функции advance(), distance(), iter_swap().
34) Итераторы стандартной библиотеки. Обратные итераторы.
35) Итераторы вставки. Разновидности итераторов вставки.
36) Потоковые итераторы. Итераторы входного и выходного потоков.
37) Строки STL (класс string). Обращение к элементам строки. Размер и емкость строки.
38) Строки STL (класс string). Операции присваивания, вставки, замены и удаления.
39) Строки STL (класс string). Подстроки и конкатенация. Поиск символов и строк. Строки в качестве контейнеров. Строки и итераторы.
40) Ввод/вывод с помощью потоковых объектов.

Целью промежуточных аттестаций по дисциплине является оценка качества освоения дисциплины обучающимися.
Промежуточная аттестация предназначена для объективного подтверждения и оценивания достигнутых результатов обучения после завершения изучения дисциплины. Осуществляется в конце семестра и представляет собой итоговую оценку знаний по дисциплине «Объектно-ориентированное программирование» в виде проведения экзамена.
Промежуточная аттестация может проводиться в устной или письменной форме. На промежуточную аттестацию отводится до 30 баллов.

Форма экзаменационного билета
по учебной дисциплине

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования «Кабардино-Балкарский государственный университет им. Х.М. Бербекова» (КБГУ)

Институт информатики, электроники и робототехники
Кафедра электроники и цифровых информационных технологий
Дисциплина – Объектно-ориентированное программирование

БИЛЕТ № 1

1. Наследование классов. Терминология, применяемая для описания иерархий наследования. Доступ к полям и методам базового класса при наследовании. Спецификаторы доступа при наследовании.

2. Проблема динамической идентификации типов. Виртуальные функции. Статическое и динамическое связывание. Виртуальный деструктор. Чистые виртуальные функции и абстрактные классы.

Руководитель ОПОП
к.т.н., доцент _____ О.А. Молоканов

Зав. кафедрой электроники
и цифровых информационных технологий,
д.т.н., профессор _____ Р.Ш. Тешев