ПРИЛОЖЕНИЕ

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«Рязанский государственный радиотехнический университет имени В.Ф. Уткина»**

КАФЕДРА ЭЛЕКТРОННЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

**ОЦЕНОЧНЫЕ МАТЕРИАЛЫ**

**«Основы алгоритмизации**

**и объектно-ориентированное программирование»**

02.03.03 «Математическое обеспечение

и администрирование информационных систем»

Квалификация (степень) выпускника – бакалавр

Форма обучения – очная, очно-заочная

Рязань

**1 ОБЩИЕ ПОЛОЖЕНИЯ**

Оценочные материалы – это совокупность учебно-методических материалов (практических заданий, описаний форм и процедур проверки), предназначенных для оценки качества освоения обучающимися данной дисциплины как части ОПОП.

Цель – оценить соответствие знаний, умений и владений, приобретенных обучающимся в процессе изучения дисциплины, целям и требованиям ОПОП в ходе проведения промежуточной аттестации.

Основная задача – обеспечить оценку уровня сформированности компетенций.

Контроль знаний обучающихся проводится в форме промежуточной аттестации.

Промежуточная аттестация проводится в форме зачета, экзамена и защиты курсовой работы. Форма проведения зачета и экзамена - тестирование, письменный опрос по теоретическим вопросам и выполнение практического задания.

**2 ОПИСАНИЕ ПОКАЗАТЕЛЕЙ И КРИТЕРИЕВ ОЦЕНИВАНИЯ КОМПЕТЕНЦИЙ**

Сформированность каждой компетенции (или ее части) в рамках освоения данной дисциплины оценивается по трехуровневой шкале:

1. пороговый уровень является обязательным для всех обучающихся по завершении освоения дисциплины;
2. продвинутый уровень характеризуется превышением минимальных характеристик сформированности компетенций по завершении освоения дисциплины;
3. эталонный уровень характеризуется максимально возможной выраженностью компетенций и является важным качественным ориентиром для самосовершенствования.

**Уровень освоения компетенций, формируемых дисциплиной:**

**Описание критериев и шкалы оценивания тестирования:**

|  |  |
| --- | --- |
| **Шкала оценивания** | **Критерий** |
| 3 балла  (эталонный уровень) | уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 85 до 100% |
| 2 балла  (продвинутый уровень) | уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 70 до 84% |
| 1 балл  (пороговый уровень) | уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 50 до 69% |
| 0 баллов | уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 0 до 49% |

**Описание критериев и шкалы оценивания теоретического вопроса:**

|  |  |
| --- | --- |
| **Шкала оценивания** | **Критерий** |
| 3 балла  (эталонный уровень) | выставляется студенту, который дал полный ответ на вопрос, показал глубокие систематизированные знания, смог привести примеры, ответил на дополнительные вопросы преподавателя |
| 2 балла  (продвинутый уровень) | выставляется студенту, который дал полный ответ на вопрос, но на некоторые дополнительные вопросы преподавателя ответил только с помощью наводящих вопросов |
| 1 балл  (пороговый уровень) | выставляется студенту, который дал неполный ответ на вопрос в билете и смог ответить на дополнительные вопросы только с помощью преподавателя |
| 0 баллов | выставляется студенту, который не смог ответить на вопрос |

**Описание критериев и шкалы оценивания практического задания:**

|  |  |
| --- | --- |
| **Шкала оценивания** | **Критерий** |
| 3 балла  (эталонный уровень) | Задача решена верно |
| 2 балла  (продвинутый уровень) | Задача решена верно, но имеются неточности в логике решения |
| 1 балл  (пороговый уровень) | Задача решена верно, с дополнительными наводящими вопросами преподавателя |
| 0 баллов | Задача не решена |

**Описание критериев и шкалы оценивания курсовой работы**

|  |  |
| --- | --- |
| **Шкала оценивания** | **Критерий** |
| Оценка «отлично»  (эталонный уровень) | Курсовая работа (КР) выполнена в полном объеме, нет замечаний по разработке алгоритмов и программ, работа выполнена самостоятельно, пояснительная записка к КР оформлена аккуратно, соблюдались сроки сдачи и защиты КР, при защите КР студент ответил на все предложенные вопросы |
| Оценка «хорошо»  (продвинутый уровень) | Курсовая работа выполнена в полном объеме, присутствуют незначительные замечания по разработке алгоритмов и программ, проект выполнен самостоятельно, пояснительная записка к КР оформлена аккуратно, соблюдались сроки сдачи и защиты КР, при защите КР студент ответил не на все предложенные вопросы (правильных ответов не менее 75%) |
| Оценка «удовлетворительно»  (пороговый уровень) | Курсовая работа выполнена в полном объеме, присутствуют ошибки при разработке алгоритмов и программ, КР выполнена самостоятельно, по оформлению пояснительной записки к КР имеются замечания, частично соблюдались сроки сдачи и защиты КР, при защите КР студент ответил не на все предложенные вопросы (правильных ответов не менее 50%) |
| Оценка «неудовлетворительно» | Курсовая работа выполнен не в полном объеме, присутствуют грубые ошибки при разработке алгоритмов и программ, КР выполнена не самостоятельно, по оформлению пояснительной записки к КР имеются замечания, не соблюдались сроки сдачи и защиты КР, при защите КР студент ответил не на все предложенные вопросы (правильных ответов менее 50%) |

На промежуточную аттестацию выносится тест, два теоретических вопроса и 1 задача. Максимально студент может набрать 12 баллов. Итоговый суммарный балл студента, полученный при прохождении промежуточной аттестации, переводится в традиционную форму по системе «отлично», «хорошо», «удовлетворительно» и «неудовлетворительно», «зачтено», «не зачтено».

**Оценка «отлично»** выставляется студенту, который набрал в сумме 12 баллов (выполнил все задания на эталонном уровне). Обязательным условием является выполнение всех предусмотренных в течение семестра практических заданий.

**Оценка «хорошо»** выставляется студенту, который набрал в сумме от 8 до 11 баллов при условии выполнения всех заданий на уровне не ниже продвинутого. Обязательным условием является выполнение всех предусмотренных в течение семестра практических заданий.

**Оценка «удовлетворительно»** выставляется студенту, который набрал в сумме от 4 до 7 баллов при условии выполнения всех заданий на уровне не ниже порогового. Обязательным условием является выполнение всех предусмотренных в течение семестра практических заданий.

**Оценка «неудовлетворительно»** выставляется студенту, который набрал в сумме менее 4 баллов или не выполнил все предусмотренные в течение семестра практические задания.

**Оценка «зачтено»** выставляется студенту, который набрал в сумме не менее 4 баллов при условии выполнения всех заданий на уровне не ниже порогового. Обязательным условием является выполнение всех предусмотренных в течение семестра практических заданий.

**Оценка «не зачтено»** выставляется студенту, который набрал в сумме менее 4 баллов или не выполнил все предусмотренные в течение семестра практические задания.

|  |  |
| --- | --- |
| **Код компетенции** | **Результаты освоения ОПОП**  **Содержание компетенций** |
| **ПК-1** | Способен проектировать программное обеспечение с использованием современных инструментальных средств |
| **ПК-1.1** Проектирует и разрабатывает программное обеспечение | |
| **ПК-1.2** Применяет современные инструментальные средства при разработке программного обеспечения | |
| **ПК-4** | Способен использовать современные методы разработки и реализации конкретных алгоритмов математических моделей на базе языков программирования и пакетов прикладных программ моделирования |
| **ПК-4.2.** Применяет пакеты прикладных программ моделирования | |

**3 ПАСПОРТ ОЦЕНОЧНЫХ МАТЕРИАЛОВ ПО ДИСЦИПЛИНЕ**

|  |  |  |
| --- | --- | --- |
| **Контролируемые разделы (темы)  дисциплины** | **Код контролируемой компетенции (или её части)** | **Вид, метод, форма оценочного мероприятия** |
| Раздел 1. Общие принципы разработки программного обеспечения | ПК-1.1, ПК-1.2 | Зачет,  курсовая работа |
| Раздел 2. Основы языка программирования С/С++ | ПК-1.1, ПК-1.2 | Зачет,  курсовая работа |
| Раздел 3. Типы, определяемые пользователем: перечислимый тип, структуры и объединения | ПК-1.1, ПК-1.2 | Зачет,  курсовая работа |
| Раздел 4. Функции с переменным числом параметров. Рекурсия. Параметры со значениями по умолчанию. Встроенные функции. Перегрузка функций. Шаблоны | ПК-1.1, ПК-1.2 | Зачет,  курсовая работа |
| Раздел 5. Файлы | ПК-1.1, ПК-1.2 | Зачет,  курсовая работа |
| Раздел 6. Указатели и динамические структуры данных | ПК-1.1, ПК-1.2 | Зачет,  курсовая работа |
| Раздел 7. Простейший графический интерфейс в Visual C++. | ПК-1.2 | Зачет,  курсовая работа |
| Раздел 8. Введение в объектно- ориентированное программирование | ПК-1.1, ПК-1.2 | Зачет  курсовая работа |
| Раздел 9. Объектно-ориентированное программирование | ПК-1.1, ПК-1.2, ПК-6.2 | Экзамен |

**4 ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ ИЛИ ИНЫЕ МАТЕРИАЛЫ**

4.1. Промежуточная аттестация в форме экзамена и зачета

|  |  |
| --- | --- |
| **Код компетенции** | **Результаты освоения ОПОП**  **Содержание компетенций** |
| ПК-1 | Способен проектировать программное обеспечение с использованием современных инструментальных средств |

**ПК-1.1. Проектирует и разрабатывает программное обеспечение**

**Типовые вопросы открытого типа:**

## 1. Каковы основные концепции ООП?

**Основными концепциями ООП являются: наследование, инкапсуляция, полиморфизм, абстракция**

## 2. Что такое инкапсуляция?

**Инкапсуляция является частью концепции ООП. Она относится к объединению данных с методами, которые работают с этими данными. Это также помогает ограничить любой прямой доступ к некоторым компонентам объекта.**

## 3. Что такое перегрузка функций (методов)?

**Существует концепция, согласно которой два или более метода могут иметь одинаковое имя. Но они должны иметь разные параметры, разное количество параметров, разные типы параметров или и то, и другое. Такие методы известны как перегруженные методы, и эта особенность называется перегрузкой методов.**

## 4. Типы наследования в ООП

**Множественное наследование. Одноуровневое наследование. Многоуровневое наследование**

5. Что такое класс?

**Класс является абстрактным типом данных, определяемым пользователем, и представляет собой модель реального объекта в виде данных и функций для работы с ними.**

6. Для чего нужен конструктор?

**Конструктор предназначен для инициализации объекта и вызывается автоматически при его создании.**

7. Что такое объект?

**Объект – это экземпляр класса, а также он обладает собственной индивидуальностью и поведением.**

8. Типы конструкторов

**Конструктор по умолчанию. Конструктор копирования. Конструктор с аргументами.**

9. Что такое модификаторы доступа?

**Модификаторы доступа или спецификаторы доступа – это ключевые слова в объектно-ориентированных языках. Они помогают установить доступность классов, методов и других членов.**

10. Зачем нужна инкапсуляция?

**Инкапсуляция позволяет скрыть данные и обернуть данные и код, который работает над ними, в единое целое.**

11. Наследование – это …

**возможность создания иерархии классов, когда потомки наследуют все свойства своих предков, могут их изменять и добавлять новые. Свойства при наследовании повторно не описываются, что сокращает объем программы**

12. \_\_\_\_\_\_\_\_\_\_ это функция, которая объявляется в базовом классе с использованием ключевого слова \_\_\_\_\_\_\_ и переопределяется в одном или нескольких производных классах

**Виртуальная функция. virtual**

## 13. Что такое полиморфизм?

**Полиморфизм – концепция, согласно которой различные классы могут использоваться с одним и тем же интерфейсом. Каждый из этих классов может иметь свою собственную реализацию интерфейса.**

## 14. Что такое статический полиморфизм?

**Статический полиморфизм или статическое связывание – это один из видов полиморфизма, который возникает во время компиляции.**

## 15. Что такое динамический полиморфизм?

**Динамический полиморфизм, динамическое связывание или полиморфизм во время выполнения – это также часть полиморфизма, который в основном реализуется во время выполнения программы.**

## 16. Что такое блок try/catch?

**Блок try/catch помогает обрабатывать исключения. Блок try объясняет набор утверждений, в которых может возникнуть ошибка. Блок catch в основном перехватывает исключение.**

## 17. Что такое тип данных?

**Тип данных однозначно определяет: диапазон возможных значений данных; допустимые действия над данными (операции и функции); внутреннее представление данных (не всегда).**

## 18. Что такое массив?

**Массив – это структурированный тип данных, в котором число элементов фиксировано при объявлении, тип компонентов массива одинаков, и имеется возможность прямого доступа к каждой отдельной компоненте объекта через индекс.**

## 19. Что такое объединение

**Объединение – это структурированный тип данных, в котором число элементов фиксировано при объявлении, тип компонентов различен, и имеется возможность прямого доступа всякий раз только к одному из элементов объединения. Все элементы объединения занимают одну и ту же область памяти.**

## 20. Основные сущности в STL:

**Три наиболее важные– это контейнеры, алгоритмы и итераторы.**

**Типовые тестовые вопросы:**

**Вопрос 1**

*Выберите верное определение алгоритма:*

1) Алгоритм – набор инструкций, описывающих порядок действий исполнителя для достижения некоторого результата.

2) Алгоритм – набор упорядоченных действий исполнителя для достижения некоторого результата.

+3) Алгоритм – набор инструкций, описывающих порядок действий компьютера для достижения некоторого результата.

4) Алгоритм – набор инструкций для написания программного кода и составления схемы алгоритма при разработке программы.

**Вопрос 2**

*Выберите все свойства алгоритма:*

+1) Массовость

2) Идейность

+3) Понятность

+4) Дискретность

5) Лаконичность

+6) Конечность

+7) Определенность

8) Уникальность

9) Работоспособность

+10) Эффективность.

**Вопрос 3**

*Что такое и зачем нужна отладка?*

1) Отладка – этап разработки компьютерной программы, на котором обнаруживают, локализуют и устраняют ошибки. Она необходима для того, чтобы пользователь мог увидеть ошибки в программе и сообщить о них разработчику.

+2) Отладка – этап разработки компьютерной программы, на котором обнаруживают, локализуют и устраняют ошибки. Она необходима для того, чтобы разработчик мог найти ошибки в программе и устранить их.

3) Отладка – этап разработки компьютерной программы, на котором отлаживают совместную работу программного обеспечения.

**Вопрос 4**

*ЕСПД – что это такое?*

+1) Единая система программной документации – комплекс государственных стандартов Российской Федерации, устанавливающих взаимосвязанные правила разработки, оформления и обращения программ и программной документации.

2) Единая система программной документации – комплекс сопроводительной программной документации для разрабатываемых программ.

3) Единая система программной документации – комплекс документации для разработчиков, в которой представлены вспомогательные материалы для программиста

**Вопрос 5**

*Что такое тип данных?*

+1) Определяет возможный диапазон значения переменных, а также операции и функции с ними

2) Определяет внешний вид значений при записи в коде программы

3) Определяет внешний вид значений, возможный диапазон и операции с ними

4) Определяет внутреннее представление данных

**Вопрос 6**

*Переменные – это:*

+1) величины, которые могут менять свое значение в процессе выполнения программы

2) величины, которые не могут менять своего значения в процессе выполнения программы

3) обозначают строки программы, на которые передается управление вовремя выполнение программы

**Вопрос 7**

*Указатель – это переменная, которая содержит в качестве своего значения \_\_\_\_\_ другой переменной.*

1) Индекс

+2) Адрес

3) Код

4) Двоичное представление

**Вопрос 8**

*Основным блоком в программе консольного приложения на языке Си является:*

1) Самая большая функция в программе

+2) Функция *main()*

3) Блок переменных и констант

**Вопрос 9**

*В языке Си оператор присваивания имеет полную и краткую формы записи. Выберите из списка ниже верное использование данного оператора.*

*+*1) *a = 10;*

*+*2) *b += 12.5;*

3) *c == 'c';*

4) *d =+ 7.123;*

**Вопрос 10**

*Необходимо найти наибольшее из двух чисел. Выберите верный способ.*

*+*1) *max = a > b ? a : b;*

2) *max = a > b : a ? b;*

3) *max = a ? b : a : b;*

**Вопрос 11**

*Какой из следующих операторов - оператор сравнения двух переменных?*

1) :=

2) equal

+ 3) ==

1. =

**Вопрос 12**

*Укажите операцию, приоритет выполнения которой больше остальных*

1) /

+2) ++

3) ()

4) \*

5) +

**Вопрос 13**

*Выберите правильный синтаксис функций:*

1) Любой\_тип имя\_функции (список параметров)

2) Возвр.\_тип имя\_функции();

3) Возвр.\_тип имя функции() {тело функции}

+4) Возвр.\_тип имя\_функции (список параметров) { тело функции}

**Вопрос 14**

*Выберите правильное утверждение для классов памяти:*

1) Область действия регистровой переменной – вся программа

2) Статическая глобальная переменная ничем не отличается от статической локальной переменной

+3) Область действия внешней переменной – вся программа

4) Область действия автоматической переменной – файл

**Вопрос 15**

*В каком заголовочном файле хранятся средства для описания и использования функций с переменным числом аргументов?*

1) stdlib.h

2) ctype.h

3) stdio.h

+4) stdarg.h

5) string.h

**Вопрос 16**

*В каких из следующих фрагментов кода верно приведена инициализация двумерного массива?*

+1) float Example1[2][2] = { {3.2, 5.6} , {1.8, 9.3} };

2) int Example2[][] = { {1, 2, 3, 4} , {5, 6, 7, 8} };

+3) int Example3[][3] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

4) float Example4[3][] = { {1.4, 5.8, 6.9} ; {7.4, 5.5, 2.4} }.

**Вопрос 17**

*Какие из следующих утверждений про объединения и структуры верны?*

1) При объявлении объединения его нельзя инициализировать;

2) Для переменных типа объединение места в памяти выделяется столько, сколько нужно элементу объединения, имеющему наименьшую длину в байтах;

+3) При объявлении можно смешивать объединения и структуры;

+4) Каждый раз только один элемент объединения доступен для обращения.

**Вопрос 18**

*Укажите правильное объявление указателя.*

1) int x;

2) ptr x;

+3) int \*x;

4) int &x;

**Вопрос 19**

*Какой из следующих записей используется операция разыменования?*

1) a;

+2) \*a;

3) address(a);

4) &a;

**Вопрос 20**

*Как правильно освободить память, после выполнения данного кода?*

char \*a; a = new char [20];

1) delete a;

2) delete a[];

+3) delete [] a;

**Вопрос 21**

*Какие функции используются для форматированного ввода-вывода?*

+1) fscanf()

2) fopen()

3) printf()

+4) fprintf()

**Вопрос 22**

*Выберете два прототипа для блочного ввода и вывода данных.*

1) int fputs(char \*s, FILE \*f);

+2) int fread(void \*ptr, int size, int n, FILE \*f);

3) int fscanf (FILE \*fp, const char \*fmt, ...);

+4) int fwrite(void \*ptr ,int size, int n, FILE \*f);

**Вопрос 23**

*Выберете правильно записанный прототип открытия файла.*

+1) FILE \*fopen(const char \*filename, const char \*mode);

2) int fprintf (FILE \*fp, char \*fmt, ...);

3) int \*fopen(const char \*filename, const char \*mode);

4) FILE \*fopen(const char \*filename);

**Вопрос 24**

*Что относится к символьному и строчному вводу- выводу из нижеперечисленных прототипов?*

1) int fwrite(void \*ptr ,int size, int n, FILE \*f);

2) int fread(void \*ptr, int size, int n, FILE \*f);

+3) int fputs(char \*s, FILE \*f);

+4) char \*fgets(char \*s, int n, FILE \*f);

5) int feof(FILE \* filename);

+6) int fputc(int c, FILE\*fp);

7) int fscanf (FILE \*fp, const char \*fmt, ...);

**Вопрос 25**

*Один из принципов объектно-ориентированного программирования:*

+ а) Инкапсуляция

б) Ингаляция

в) Инструкция

**Вопрос 26**

*Один из принципов объектно-ориентированного программирования:*

а) Отдача

б) Передача

+ в) Наследование

**Вопрос 27**

*Свойство объектов, при котором действие с одинаковыми именами вызывает различное поведение для различных объектов:*

+ а) Полиморфизм

б) Передача

в) Монорфизм

**Вопрос 28**

*Действие, которое может выполнить объект:*

+ а) Метод

б) Событие

в) Свойство

**Вопрос 29**

*Какая функция, не будучи компонентом класса, имеет доступ к его защищенным и внутренних компонентов:*

а) Статическая

+ б) Дружественная

в) Шаблонная

**Вопрос 30**

*Какая из перечисленных функций не может быть конструктором:*

а) String (const int a)

б) String (String & s)

+ в) void String ()

**Вопрос 31**

*Примеры полиморфизма в С++:*

+а) перегрузка функций

+б) шаблоны функций

в) использование деструкторов

+г) механизм виртуальных методов

д) значения по умолчанию

**Вопрос 32**

*Отметьте модификаторы, для которых характерна доступность из любого места программы*

+а) public

б) private

в) protected

**Вопрос 33**

*Отметьте модификаторы, для которых характерна доступность производного класса*

+а) public

б) private

+в) protected

**Вопрос 34**

*Отметьте модификаторы, для которых характерна доступность из самого класса*

+а) public

+б) private

+в) protected

**Вопрос 35**

*Отметьте последовательные контейнеры STL*

+а) векторы

б) мультиотображения

+в) очереди с двусторонним доступом.

г) множества

д) стек

е) мультимножества

ж) приоритетная очередь

з) отображения

и) очередь

+к) списки

**Вопрос 36**

*Отметьте ассоциативные контейнеры STL*

а) векторы

+б) мультиотображения

в) очереди с двусторонним доступом.

+г) множества

д) стек

+е) мультимножества

ж) приоритетная очередь

+з) отображения

и) очередь

к) списки

**Вопрос 37**

*Отметьте специализированные контейнеры STL (адаптеры когтейтеров)*

а) векторы

б) мультиотображения

в) очереди с двусторонним доступом.

г) множества

+д) стек

е) мультимножества

+ж) приоритетная очередь

з) отображения

+и) очередь

к) списки

**ПК-1.2. Применяет современные инструментальные средства при разработке программного обеспечения**

**Типовые вопросы открытого типа:**

1. Что такое форма?

**Форма — это главный контейнер, в котором размещаются компоненты самой среды. С помощью этих компонентов и реализуется конкретный алгоритм определенной задачи.**

1. Какие типы компонентов вы знаете?

**Различают два типа компонентов: визуальные и невизуальные.**

1. Визуальный компонент – это …

**элемент пользовательского интерфейса, например, поле редактирования (TextBox), поле отображения текста (Label), кнопка (Button).**

1. Визуальные компоненты отображаются …

**как на форме (во время разработки программы), так и в окне программы во время ее работы.**

1. Невизуальные компоненты отображаются …

**только на форме во время разработки программы.**

1. Компоненты, которые программист может использовать при разработке программ, в среде Visual Studio находятся …

**на вкладке** **Toolbox**.

1. Каждый компонент характеризуется наборами данных, определяющими его функциональность: …

**свойства, события и методы.**

1. Свойства – это…

**переменные, которые влияют на состояние объекта. Например, ширина, высота, положение кнопки на форме или надпись на ней.**

1. Методы – это …

**функции, то есть это то, что объект умеет делать (вычислять).**

1. События – это …

**функции, которые вызываются при наступлении определенного события. Например, пользователь нажал на кнопку, вызывается процедура обработки этого нажатия.**

1. \_\_\_\_\_\_\_\_\_\_\_\_\_– комплекс программных средств, используемый программистами для разработки программного обеспечения (ПО).

**Интегрированная среда разработки (далее IDE)**

1. Что включает в себя среда разработки?

**текстовый редактор, компилятор и/или интерпретатор, средства автоматизации сборки, отладчик**

1. Какие основные пункты входят в главное меню Visual Studio?

**Меню «Файл», меню «Правка», меню «Вид», меню «Построение», меню «Отладка»**

1. Какие команды содержит меню «Файл»?

**команды для создания, открытия и сохранения файла или проекта, имеется возможность открыть последние из ранее открытых, напечатать текущую страницу.**

**Типовые тестовые вопросы:**

**Вопрос 1**

*Компоненты, которые видны во время работы приложения, с ними напрямую может взаимодействовать пользователь, называются:*

а) Абстрактными

б) Видимыми

+ в) Визуальными

**Вопрос 2**

*Этот компонент в Visual Studio (в VS) предназначен для вывода текста на поверхность формы:*

+ а) Label

б) Edit

в) Button

**Вопрос 3**

*Что обозначает свойство Text у формы в VS*

+а) Название формы

б) текст на форме

в) скрытый текст на форме

**Вопрос 4**

*Компоненты, которые не видны во время работы приложения называются:*

а) Скрытими

б) Невидимыми

+ в) Невизуальными

**Вопрос 4**

*Свойство Name в VS отвечает за:*

а) Название компонента

+ б) Имя компонента

в) Назначение компонента

**Вопрос 5**

*Имя формы в VS, используется для управления формой и доступа к компонентам формы:*

+ а) Свойство формы Name

б) Значение формы Name

в) Следствие формы Name

**Вопрос 6**

*Компонент, представляющий собой поле ввода-редактирования строки символов в VS:*

а) Memo

б) Edit

в) CheckBox

+г) TextBox

**Вопрос 7**

*Что обозначает свойство CheckAllign компонента RadioButton в VS?*

а) Выбор переключателя

+б) Положение переключателя в поле компонента.

В) Номер переключателя

**Вопрос 8**

*Что указывается минимально в методе MessgeBox::Show() в VS?*

а) Имя формы

б) Кнопки на форме

+в) Строка-информации

г) Иконка

д) Имя метода

**Вопрос 9**

*Что собой представляет компонент Panel в VS?*

а) форма внутри главной формы

+б) контейнер для других компонентов

в) фон главной формы

**Вопрос 10**

*Какой компонент позволяет работать с таблицей в VS?*

а) Table

+б) DataGridView

в) DataTableView

**Вопрос 11**

*Какие команды находятся в меню «Правка»?*

а) отладка,

+б) копирование,

+в) вставка,

+г) удаление,

+д) поиск,

+е) замена,

ж) компиляция

**Вопрос 12**

*Назначение меню «Вид»?*

+а) позволяет отобразить те или иные панели в среде

б) изменяет размер шрифта

в) изменяет язык IDE

**Вопрос 13**

*Назначение меню «Построение»?*

а) Строит диаграмму классов

+б) содержит основные действия, связанные с компиляцией

в) Создает форму

**Вопрос 14**

*Назначение меню «Отладка»?*

+а) управляет отладкой проекта

+б) управляет запуском проекта

в) компилирует программу

г) выводит список синтаксических ошибок

|  |  |
| --- | --- |
| **ПК-6** | Способен использовать современные методы разработки и реализации конкретных алгоритмов математических моделей на базе языков программирования и пакетов прикладных программ моделирования |
| **ПК-6.2.** Применяет пакеты прикладных программ моделирования | |

**Типовые вопросы открытого типа:**

1. Что такое UML?

**UML(унифицированный язык моделирования) — язык** [**графического**](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B7%D1%83%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) **описания для** [**объектного моделирования**](https://ru.wikipedia.org/w/index.php?title=%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&action=edit&redlink=1) **в области** [**разработки программного обеспечения**](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F)**, для** [**моделирования бизнес-процессов**](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%B1%D0%B8%D0%B7%D0%BD%D0%B5%D1%81-%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D0%B2)**,** [**системного проектирования**](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) **и отображения** [**организационных структур**](https://ru.wikipedia.org/wiki/%D0%9E%D1%80%D0%B3%D0%B0%D0%BD%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0)

1. Какие типы UML диаграмм вы знаете?

**Структурные диаграммы и диаграммы поведения.**

1. Что такое диаграмма классов?

**Диаграмма классов (Class diagram) — статическая структурная диаграмма, описывающая структуру системы, демонстрирующая классы системы, их атрибуты, методы и зависимости между классами**

1. Как отображается класс на диаграмме?

**В виде прямоугольника разделенного на 3 части:**

* **Имя класса**
* **Атрибуты класса, тип которых записывается через двоеточие**
* **Методы. Тип, который возвращает метод записывается через двоеточие в самом конце сигнатуры метода**

1. Как на диаграмме классов отображаются модификаторы области видимости?

**«+» - public, «-» - private, «#» - protected**

1. Каждый параметр в методе может иметь описание направленности в методе. Это …

**in, out, inout**

1. Перечислите межклассовые соотношения

**Ассоциация, наследование, реализация, зависимость, агрегация, композиция**

1. Что такое агрегация?

**Агрегация – это особый вид отношения между классами, когда один класс является частью другого, но они могут жить и по отдельности**

1. Что такое композиция?

**Композиция – это разновидность агрегации, только в этом случае классы являющиеся частью другого класса уничтожаются когда уничтожается класс агрегатор.**

**Типовые тестовые вопросы:**

**Вопрос 1**

*Какая информация о классе отображается на диаграмме классов*

+а) имя класса

+б) атрибуты класса

в) указывается, что это производный класс

+г) методы класса

д) отмечается, что это базовый класс

**Вопрос 2**

*Верно ли следующее утверждение:*

*Агрегация – это особый вид отношения между классами, когда один класс является частью другого класса и который уничтожается когда уничтожается класс агрегатор.*

а) Да

+б) Нет

**Вопрос 3**

*Что обозначает символ «+» на диаграмме классов?*

+а) модификаторы области видимости public

б) модификаторы области видимости private

в) модификаторы области видимости protected

г) операция сложения

**Вопрос 4**

*Что обозначает символ «-» на диаграмме классов?*

а) модификаторы области видимости public

+б) модификаторы области видимости private

в) модификаторы области видимости protected

г) операция вычитания

**Вопрос 5**

*Что обозначает символ «#» на диаграмме классов?*

а) модификаторы области видимости public

б) модификаторы области видимости private

+в) модификаторы области видимости protected

г) операция сложения или вычитания

**Вопрос 6**

*Отметьте межклассовые соотношения*

+а) ассоциация,

+б) наследование,

в) выполнение,

г) отторжение,

+д) агрегация,

+е) композиция

**Вопрос 7**

*Какие диаграммы относятся к структурным?*

+а) Диаграмма классов

+б) Диаграмма пакетов

в) Диаграмма деятельности

г) Диаграмма состояний

4.2. Промежуточная аттестация в форме курсовой работы

|  |  |
| --- | --- |
| **Код компетенции** | **Результаты освоения ОПОП**  **Содержание компетенций** |
| **ПК-1** | Способен проектировать программное обеспечение с использованием современных инструментальных средств |
| **ПК-1.1** Проектирует и разрабатывает программное обеспечение | |
| **ПК-1.2** Применяет современные инструментальные средства при разработке программного обеспечения | |

**Типовое задание для курсовой работы по дисциплине:**

Целью курсового проектирования являются разработка и отладка приложения с использованием динамических структур данных, написанного на алгоритмическом языке программирования С++ в среде визуального программирования Visual C++.

Тема курсовой работы: ««Разработка приложения с использованием динамических структур данных».

Курсовая работа (КР) представляет собой самостоятельную работу по заданной теме. Работа предполагает:

* домашнюю внеаудиторную подготовку;
* выполнение практической части дома или в классе персональных компьютеров на кафедре;
* консультации по КР;
* предъявление промежуточных результатов для проверки и контроля хода курсового проектирования;
* написание и оформление пояснительной записки;
* сдачу и защиту КР в сроки согласно учебному графику.

**Исходные данные**

* Тип компьютера IBM РC совместимый.
* Операционная система Windows XP (не ниже).
* Язык программирования C/C++.
* Среда программирования: Microsoft Visual Studio.
* Текст задания согласно варианту.

**Общие требования**

Все варианты заданий связаны с разработкой таблицы данных с использованием линейных однонаправленных списков.

Разрабатываемая программа должна обязательно выполнять следующие запросы:

* заполнение пустой таблицы (создание списка);
* сохранение таблицы (списка) в файле;
* чтение таблицы (списка) из файла;
* вывод таблицы на экран;
* добавление элементов в таблицу (в список);
* удаление элементов из таблицы (из списка);
* а также все запросы, которые указаны в индивидуальном задании.

Вызовы запросов должны осуществляться через систему меню с использованием средств визуального программирования. Необходимо предусмотреть контроль ошибок пользователя при вводе данных. Результаты некоторых запросов (по согласованию с преподавателем на этапе уточнения технического задания) должны выводиться в виде графиков или диаграмм. Приложение обязательно должно иметь заставку.

Все элементарные действия (заполнение списка, запись элемента в список и т.д.) должны быть оформлены в виде подпрограмм, а все (или некоторые) объявления и подпрограммы должны быть оформлены в виде модуля (модулей).

***Перечень индивидуальных заданий:***

**Вариант № 1. Абоненты библиотеки**

Информация об абонентах библиотеки следующая:

* номер читательского билета;
* ФИО;
* год рождения;
* пол;
* подразделение (кафедра, номер группы);
* должность;
* отметка о перерегистрации;
* имеются книги на срок;
* дата возврата книг.

Написать программу, которая выполняет следующие запросы:

* вывод информации об абоненте по номеру читательского билета;
* упорядочение таблицы по ФИО;
* вывод списка абонентов-должников определенного факультета;
* вывод списка абонентов, которые не прошли перерегистрацию;
* вывод процентного соотношения сотрудников и студентов среди абонентов.

**Вариант № 2. Анкета студента**

Анкета студента имеет следующие пункты:

* ФИО;
* пол;
* факультет;
* номер группы;
* адрес постоянного проживания;
* сведения о получении стипендии;
* вид спорта.

Написать программу, которая выполняет следующие запросы:

* вывод списка студентов определенной группы по алфавиту;
* вывод списка студентов на факультете, которые занимаются спортом;
* вывод списка иногородних студентов на факультете;
* вывод списка студентов на факультете, которые не получают стипендию;
* вывод процентного соотношения мужчин и женщин на факультете.

**Вариант № 3. Анкета абитуриента**

Анкета абитуриента имеет следующие пункты:

* ФИО;
* адрес постоянного проживания;
* льгота (инвалидность, сирота, целевой набор);
* баллы по ЕГ (математика, физика, русский язык);
* номер направления. По каждой специальности определен проходной балл.

Написать программу, которая выполняет следующие запросы:

* упорядочение таблицы по ФИО абитуриента;
* определение процента иногородних абитуриентов;
* вывод списка абитуриентов, которые поступили на определенное направление;
* вывод номеров направлений в порядке убывания «популярности»;
* вывод процентного соотношения льготников и абитуриентов, которые поступают в вуз на общих основаниях.

Полный список индивидуальных заданий находится в методических указаниях к КР. Дублирование тем для индивидуального исследования в пределах одной учебной группы не допускается.

Защита курсовой работы назначается по итогам проверки предоставленной пояснительной записки, оформленной в соответствии с требованиями и разработанного приложения. Защита осуществляется в форме ответов на вопросы преподавателя.

**Типовые вопросы на защите курсовой работы:**

1. Как разрабатывался графический интерфейс для вашего приложения?
2. Какие визуальные и не визуальные компоненты вы использовали?
3. Есть ли в вашем приложении модальные окна? Для каких целей вы их используете?
4. Как создавалась заставка для вашего приложения?
5. Как создается однонаправленный динамический список?
6. Как в списке происходит поиск данных?
7. Как отображаются в вашем приложении табличные данные?
8. Как осуществляется чтение и запись данных
9. Опишите назначение компонента MainMenu. Как вы используете эту компоненту в проекте?
10. Объясните алгоритм(ы) различных функций.
11. Как эти алгоритмы реализуются на языке C++.

**Типовые теоретические вопросы для зачета по дисциплине (семестр 1)**

1. Метод проектирования программных средств. Основные этапы решения задач на ЭВМ.
2. Понятия алгоритма, его свойства. Способы представления алгоритмов. Базовые алгоритмические структуры: следование, разветвление, повторение. Способы их изображения.
3. Структура языка: алфавит, синтаксис языка. Лексическая структура языка. Ключевые слова. Идентификаторы. Константы. Комментарии.
4. Структура программы. Стиль записи программы на С/С++.
5. Понятие и классификация типов данных. Скалярные (простые) типы данных: целые, символьные, вещественные, логические. Определение и инициализация переменных. Ключевое слово typedef.
6. Операции в С/С++: присваивания, арифметические, логические, операции сдвига, сравнения, побитовые, условная, sizeof, запятая. Приоритеты операций и их направленность выполнения.
7. Выражение. Приоритет операций в выражении.
8. Явное преобразование типов. Неявное преобразование типов: расширение типа в выражении; преобразование при присваивании.
9. Понятие потока. Функции ввода и вывода данных.
10. Разветвляющиеся алгоритмы. Оператор выбора switch и команда ветвления if else. Запись команды выбора с помощью команд ветвления.
11. Операторы циклов. Использование шаблонов при создании циклов. Операторы прерывания циклов.
12. Классификация структурированных типов. Одномерные и многомерные массивы. Объявление и инициализация.
13. Указатели. Объявление и инициализация. Указатели и массивы. Операции с указателями.
14. Определение, описание и вызов функций. Тип функции. Оператор return.
15. Понятие фактических и формальных параметров. Прототипы функции.
16. Способы передачи параметров в функцию. Передача массивов в функцию. Ссылка.
17. Понятие блока. Область действия и область видимости глобальных и локальных переменных.
18. Классы памяти.
19. Техника многомодульного программирования
20. Указатели типа void. Преобразование типа указателя.
21. Указатели на указатели. Указатели и многомерные массивы. Массивы указателей.
22. Функции, возвращающие указатели. Указатели на функции.
23. Интерпретация сложных описаний.
24. Аргументы функции main().
25. Строки: определение, инициализация, функции для работы со строками.

**Типовые теоретические вопросы для зачета по дисциплине (семестр 2)**

1. Перечислимый тип enum.
2. Структуры. Определение и инициализация. Создание таблиц: массивы структур и структуры содержащие массивы. Вложенные структуры. Указатели как поля структур. Указатели на структуры и передача структур в функцию. Битовые поля структур.
3. Объединения. Определение и инициализация.
4. Файлы. Основные определения.
5. Основные этапы работы с файлом.
6. Символьный, форматированный и строковый ввод и вывод.
7. Динамическое распределение памяти в стилях С и С++.
8. Одномерные динамические массивы в стилях Си и С++.
9. Динамические структуры данных. Список.
10. Динамические структуры данных. Стек.
11. Двумерные динамические массивы в стилях Си и С++.
12. Типичные ошибки при работе с динамической памятью.
13. ООП. Основные свойства ООП.
14. Описание класса. Выделение памяти под объекты.
15. Конструкторы. Конструкторы без параметров и с параметрами. Конструктор копирования. Деструкторы.
16. Определение методов класса вне класса. Объекты в качестве аргументов. Объекты, возвращаемые функцией.
17. Указатель this.
18. Рекурсия.
19. Встроенные функции.
20. Макрофункции.
21. Параметры со значениями по умолчанию
22. Функции с переменным числом аргументов

**Типовые теоретические вопросы для экзамена по дисциплине (семестр 3)**

1. ООП. Основные свойства. Классы и объекты (определение и использование класса, вызов методов класса). Конструкторы. Перегруженные конструкторы. Конструктор копирования по умолчанию. Деструкторы.
2. Определение методов класса вне класса. Объекты в качестве аргументов функций. Объекты, возвращаемые функцией. Классы, объекты и память. Статические и константные элементы классов
3. Перегрузка операций. Перегрузка унарных операций. Перегрузка бинарных операций. Операции арифметического присваивания.
4. Наследование. Понятие базового и производного классов. Доступ к базовому классу. Спецификаторы доступа. Неизменность базового класса. Конструкторы производного класса. Перегрузка функций для производного класса. Операция разрешения.
5. Иерархия классов. Общее и частное наследование. Уровни наследования. Множественное наследование. Неопределенность при множественном наследовании.
6. Указатели и ссылки на производные типы. Виртуальные функции. Виртуальные деструкторы. Наследование виртуальных функций.
7. Чисто виртуальные функции и абстрактные классы. Сравнение раннего связывания с поздним. Виртуальные базовые классы.
8. Дружественный функции. Дружественные функции как мосты между классами. Дружественные классы.
9. Шаблонные функции. Явно заданная перегрузка обобщенной функции. Перегрузка шаблона функции. Использование стандартных параметров в шаблонных функциях. Ограничения при использовании обобщенных функций.
10. Шаблоны классов. Контекстозависимое имя класса. Создание обобщенного класса безопасного массива. Использование в обобщенных классах аргументов, не являющихся типами. Использование в шаблонных классах аргументов по умолчанию. Явно задаваемые специализации классов.
11. Основы обработки исключительных ситуаций.
12. Класс string.
13. Файловый ввод-вывод в стиле С++. Работа с текстовыми файлами. Неформатированный ввод-вывод данных в двоичном режиме. Произвольный доступ. Проверка статуса ввода-вывода
14. Стандартная библиотека шаблонов STL. Состав, основные понятия и определения.
15. Последовательные контейнеры. Вектор. Основные операторы и методы.
16. Последовательные контейнеры. Список и очереди с двусторонним доступом. Работа с контейнерами.
17. Итераторы. Недостатки обычных указателей. Категории итераторов и их возможности. Работа с итераторами.
18. Типы итераторов, поддерживаемые контейнерами. Итераторы и алгоритмы. Работа с итераторами.
19. Специализированные итераторы. Работа с итераторами.
20. Ассоциативные контейнеры. Отображение и мультиотображение. Работа с контейнерами.
21. Ассоциативные контейнеры. Множества и мультимножества. Работа с контейнерами.
22. Функциональные объекты. Предопределенные функциональные объекты. Примеры применения.