**Карта тестовых заданий**

**Компетенция:** {{ компетенция }}

**Индикатор:** {{ индикатор }}

**Дисциплина**: Межплатформенное програмиирование

**Описание теста:**

1. Тест состоит из 70 заданий, которые проверяют уровень освоения компетенций обучающегося. При тестировании каждому обучающемуся предлагается 30 тестовых заданий по 15 открытого и закрытого типов разных уровней сложности.

2. За правильный ответ тестового задания обучающийся получает 1 условный балл, за неправильный ответ – 0 баллов. По окончании тестирования, система автоматически определяет «заработанный итоговый балл» по тесту, согласно критериям оценки

3 Максимальная общая сумма баллов за все правильные ответы составляет – 100 баллов.

4. Тест успешно пройден, если обучающийся правильно ответил на 70% тестовых заданий (61 балл).

5. На прохождение тестирования, включая организационный момент, обучающимся отводится не более 45 минут. На каждое тестовое задание в среднем по 1,5 минуты.

6. Обучающемуся предоставляется одна попытка для прохождения компьютерного тестирования.

**Кодификатором** теста по дисциплине является раздел рабочей программы «4. Структура и содержание дисциплины (модуля)»

**Комплект тестовых заданий**

**Задания закрытого типа**

**Задания альтернативного выбора**

*Выберите* ***один*** *правильный ответ*

Простые (1 уровень)

1 Объявление переменной в C# имеет следующий синтаксис

А) var x \= value;

Б) x value;

В) int x \= value;

Г) x\: value;

2 Оператор "using" в C# предназначен для того чтобы

А) распорядиться объектом и освободить его ресурсы

Б) включить пространство имен в текущий файл

В) открыть файл для чтения или записи

Г) создать экземпляр класса

3 Создание класса в C# имеет следующий синтаксис

А) class MyClass

Б) function MyClass

В) struct MyClass

Г) object MyClass

4 Разница между циклами "while" и "do-while" в C# заключается в том что \_\_\_

А) цикл while проверяет условие перед выполнением цикла, а цикл do-while выполняет цикл до проверки условия

Б) цикл while можно использовать только с массивами, а цикл do-while - с любым типом данных

В) цикл while быстрее цикла do-while

Г) разницы нет, они взаимозаменяемы

5 Оператор "is" в C# предназначен для того чтобы

А) проверить, является ли объект экземпляром определенного класса или интерфейса

Б) проверить, равны ли две переменные

В) проверить, не является ли переменная нулевой

Г) проверить, имеет ли переменная определенный тип данных

Средне-сложные (2 уровень)

6 Создание массива в C# имеет следующий синтаксис

А) int[] myArray \= new int[5];

Б) myArray \= \{1, 2, 3, 4, 5\};

В) array myArray \= \{1, 2, 3, 4, 5\};

Г) int myArray[5] \= \{1, 2, 3, 4, 5\};

7 Создание исключения в C# имеет следующий синтаксис

А) throw new Exception();

Б) raise Exception();

В) throw Exception();

Г) new Exception();

8 Создание делегата в C# происходит следующим образом

А) делегат MyDelegate();

Б) делегат MyDelegate();

В) функция MyDelegate();

Г) событие MyDelegate();

9 Оператор "as" в C# предназначен для того чтобы

А) проверка того, является ли объект экземпляром определенного класса или интерфейса

Б) приведение объекта к определенному типу данных

В) преобразование строки в число

Г) проверка равенства двух переменных

10 Разница между циклами "while" и "do-while" в C# заключается в том что

А) break выходит из цикла, continue пропускает текущую итерацию

Б) break пропускает текущую итерацию, continue выходит из цикла

В) и break, и continue выходят из цикла

Г) и break, и continue пропускают текущую итерацию

11 Создание общего класса в C# происходит следующим образом

А) class MyClass<T>

Б) class MyClass<тип T>

В) class MyClass

Г) class MyClass<generic T>

12 Оператор "lock" в C# предназначен для

А) блокировки потока и предотвращения доступа других потоков к общему ресурсу

Б) того чтобы разблокировать поток и разрешить другим потокам доступ к общему ресурсу

В) создания нового потока

Г) завершения потока

13 Разница между ключевым словом "var" и явными типами данных в C# заключается в том что

А) var - это ключевое слово, которое указывает компилятору на вывод типа данных, в то время как явные типы данных определяются разработчиком

Б) var используется для переменных, тип которых неизвестен во время компиляции, а явные типы данных используются для переменных, тип которых известен во время компиляции

В) var используется только для примитивных типов данных, тогда как явные типы данных могут использоваться для всех типов данных

Г) разницы нет, var и явные типы данных могут использоваться взаимозаменяемо

14 Определение свойства в C# имеет следующий синтаксис

А) int MyProperty \{ get; set; \}

Б) property int MyProperty \{ get; set; \}

В) MyProperty \{ get; set; \}

Г) int getMyProperty() \{ \} int setMyProperty(value) \{ \}

15 Разница между блоком "try-catch" и блоком "try-finally" в C# заключается в том что

А) try-catch используется для обработки ошибок, а try-finally - для задач очистки

Б) -catch обрабатывает исключения и продолжает выполнение, а try-finally только запускает блок finally и продолжает выполнение

В) try-catch выполняет блок catch и выходит из метода, а try-finally только выполняет блок finally и выходит из метода

Г) try-catch и try-finally взаимозаменяемы и выполняют одно и то же действие

16 Создание структуры в C# имеет следующий синтаксис

А) struct MyStruct \{ \}

Б) class MyStruct \{ \}

В) object MyStruct \{ \}

Г) MyStruct \{ \}

17 Создание перечисления в C# имеет следующий синтаксис

А) enum MyEnum \{ \}

Б) MyEnum \{ \}

В) enum MyEnum \{ Value1, Value2, Value3 \}

Г) MyEnum \{ Value1, Value2, Value3 \}

18 Разница между "абстрактными" и "виртуальными" методами в C# заключается в том, что

А) абстрактные методы не могут иметь реализации, в то время как виртуальные методы могут иметь реализацию

Б) виртуальные методы не могут иметь реализацию, в то время как абстрактные методы могут иметь реализацию

В) абстрактные методы переопределяются по умолчанию, а виртуальные - нет

Г) виртуальные методы переопределяются по умолчанию, а абстрактные – нет

19 Разница между параметрами "out" и "ref" в C# заключается в том, что

А) параметры out должны быть инициализированы перед передачей в метод, параметры ref - нет

Б) параметры ref должны быть инициализированы перед передачей в метод, параметры out - нет

В) параметры out могут использоваться только с типами значений, параметры ref могут использоваться с любыми типами

Г) параметры ref могут использоваться только с типами значений, параметры out могут использоваться с любым типом

20 Разница между переменными "const" и "readonly" в C#

А) переменные const могут быть установлены только во время компиляции, в то время как переменные readonly могут быть установлены во время выполнения программы

Б) переменные readonly могут быть установлены только во время компиляции, в то время как переменные const могут быть установлены во время выполнения программы

В) переменные const можно использовать с любым типом данных, а переменные readonly - только со ссылочными типами

Г) переменные readonly могут использоваться с любым типом данных, а переменные const - только со ссылочными типами

21 Разница между " static" и "instance" методами в C#

А) статические методы могут быть вызваны без создания экземпляра класса, в то время как методы экземпляра требуют наличия экземпляра

Б) методы экземпляра могут быть вызваны без создания экземпляра класса, в то время как статические методы требуют наличия экземпляра

В) статические методы можно использовать только со статическими переменными, тогда как методы экземпляра можно использовать с переменными любого типа

Г) методы экземпляра могут использоваться только с переменными экземпляра, в то время как статические методы могут использоваться с переменными любого типа

22 Термин "межплатформенное программирование" обозначает

А) Программирование для разных операционных систем и платформ

Б) Программирование на нескольких языках программирования

В) Программирование на одной платформе

Г) Программирование только для мобильных устройств

Сложные (3 уровень)

23 Инструмент, который позволяет разрабатывать межплатформенные приложения с использованием C#

А) Xamarin

Б) Android Studio

В) Swift

Г) Eclipse

24 Языки программирования, которые могут быть использованы в межплатформенной разработке

А) Различные языки программирования

Б) Только C\#

В) Только Java

Г) Только Python

25 API в контексте межплатформенного программирования это

А) Программный интерфейс приложения

Б) Аппаратный интерфейс устройства

В) Программа для мультимедийных приложений

Г) Обновление операционной системы

*Выберите* ***два или более*** *правильных ответов*

**Задания на установление соответствия**

*Установите соответствие между левым и правым столбцами.*

Простые (1 уровень)

26 Установите соответствие

::26 Задание::

|  |  |
| --- | --- |
| 1 Процесс отображения графики на экране или печати | А) Рендеринг |
| 2 Специфический период времени, в течение которого компонент активен и может реагировать на события | Б) Жизненный цикл компонента |
| 3 Процесс адаптации приложения под различные региональные и языковые особенности | В) Локализация |
| 4 Уникальный идентификатор, используемый для аутентификации при взаимодействии с внешними сервисами | Г) API ключ |
|  | Д) Интеграция |

27 Установите соответствие

::27 Задание::

|  |  |
| --- | --- |
| 1 Позволяет создавать объекты без указания конкретных классов создаваемых объектов | А) Фабричный метод |
| 2 Определяет зависимость "один ко многим" между объектами таким образом, что при изменении состояния одного объекта все зависящие от него оповещаются и обновляются автоматически | Б) Наблюдатель |
| 3 Определяет семейство алгоритмов, инкапсулирует каждый из них и делает их взаимозаменяемыми | В) Стратегия |
| 4 Позволяет добавлять новое поведение или состояние объектам во время выполнения | Г) Декоратор |
|  | Д) Инкремент |

Средне-сложные (2 уровень)

28 Установите соответствие

::28 Задание::

|  |  |
| --- | --- |
| 1 Xamarin | А) Разработка мобильных приложений на языке C\# |
| 2 React Native | Б) Кросс-платформенная библиотека для разработки интерфейсов на JavaScript и React |
| 3 Flutter | В) Фреймворк для создания мобильных приложений с использованием языка Dart |
|  | Г) Фреймворк для разработки графических интерфейсов на языке Java |

29 Установите соответствие

::29 Задание::

|  |  |
| --- | --- |
| 1 API | А) Интерфейс, предоставляемый программой для взаимодействия с другими программами |
| 2 Сборка мусора | Б) Процесс автоматического освобождения неиспользуемой памяти в программе |
| 3 Полиморфизм | В) Возможность объектов разных классов использовать одинаковые методы или свойства |
|  | Г) Возможность выполнения асинхронных операций в программе |

30 Установите соответствие

::30 Задание::

|  |  |
| --- | --- |
| 1 async | А) Модификатор метода, указывающий, что он является асинхронным |
| 2 await | Б) Метод, позволяющий асинхронно ожидать завершения выполнения асинхронного метода |
| 3 Task | В) Объект, представляющий асинхронную операцию |
|  | Г) Асинхронная операция, возвращающая результат |

31 Установите соответствие

::31 Задание::

|  |  |
| --- | --- |
| 1 Целочисленный тип данных, хранящий значения от -2,147,483,648 до 2,147,483,647 | А) int |
| 2 Тип данных для хранения текстовых строк | Б) string |
| 3 Логический тип данных, принимающий значения true или false | В) bool |
|  | Г) double |

32 Установите соответствие

::32 Задание::

|  |  |
| --- | --- |
| 1 Thread | А) Позволяет создать новый поток выполнения в приложении |
| 2 Task | Б) Предоставляет механизмы для асинхронного выполнения операций и возврата результатов |
| 3 ThreadPool | В) Пул потоков, предоставляющий группу потоков для выполнения задач |
|  | Г) Специальные ключевые слова, упрощающие асинхронное программирование в C\# |

33 Установите соответствие

::33 Задание::

|  |  |
| --- | --- |
| 1 Portable Class Libraries (PCL) | А) Позволяют определить общий код для использования в разных платформах |
| 2 Shared Projects | Б) Позволяют создавать платформоспецифичный код для каждой целевой платформы |
| 3 .NET Standard | В) Устанавливают стандарты совместимости для библиотек в различных платформах |
|  | Г) Обеспечивают возможность реализации платформоспецифичного кода в проекте |

34 Установите соответствие

::34 Задание::

|  |  |
| --- | --- |
| 1 Кросс-платформенность | А) Способность приложения работать на разных операционных системах и устройствах без изменения кода |
| 2 Реиспользование кода | Б) Использование общего кода для разработки приложений для различных платформ |
| 3 Адаптивный интерфейс | В) Создание интерфейса, который адаптируется к разным размерам экранов и устройствам |
|  | Г) Инструменты, которые позволяют разрабатывать приложения для нескольких платформ |

Сложные (3 уровень)

35 Установите соответствие

::35 Задание::

|  |  |
| --- | --- |
| 1 MVC (Model-View-Controller) | А) Разделяет приложение на три компонента\: модель, представление и контроллер |
| 2 MVVM (Model-View-ViewModel) | Б) Предоставляет отдельный объект (ViewModel) для управления представлением |
| 3 MVP (Model-View-Presenter) | В) Разделяет приложение на три компонента\: модель, представление и презентер |
|  | Г) Схема управления состоянием приложения, в которой данные однонаправленно потоки от источника к представлению |

**Задания открытого типа**

**Задания на дополнение**

*Напишите пропущенное слово.*

Простые (1 уровень)

36 Язык \_\_\_ - это современный объектно-ориентированный язык программирования, используемый для разработки широкого спектра приложений, включая настольные приложения, веб-приложения, мобильные приложения, игры и многое другое

37 Неявно типизированная переменная, ее тип определяется компилятором во время компиляции на основе выражения, используемого для ее инициализации

38 Динамически типизированная переменная, ее тип определяется во время выполнения, что делает ее более гибкой, но и менее безопасной

39 Оператор \_\_\_ в C# используется для указания блока кода, которому требуется определенный ресурс, например, файл или соединение с базой данных. Ресурс автоматически утилизируется после завершения выполнения блока кода

40 Исключения в C# можно перехватывать. Код, который может вызвать исключение, помещается в блок \_\_\_, а код обработки исключения – в блок \_\_\_. (Перечислить ответы по очереди через пробел)

41 \_\_\_ - это коллекция элементов фиксированного размера одного типа

42 \_\_\_ - это динамически изменяемая коллекция элементов, элементы которой могут быть любого типа

Средне-сложные (2 уровень)

43 При помощи \_\_\_ в C# можете создать новый класс, который является производным от существующего класса, который содержит в себе все члены базового класса, а также может добавлять новые члены или переопределять существующие

44 В C# схема классов, определяющая набор методов, свойств и событий называется \_\_\_\_\_ . Используется для определения общего набора моделей поведения, которые могут быть реализованы несколькими классами, независимо от деталей их реализации

45 Тип в C#, который используется для передачи методов в качестве аргументов другим методам, что позволяет динамически вызывать методы и обрабатывать события называется \_\_\_\_.

46 \_\_\_ в C# - это тип значений, который представляет собой именованный постоянный набор значений. Используется для объявления набора именованных постоянных значений, что делает код более читабельным и удобным для сопровождения

47 \_\_\_ в C# - это специальный метод, который выполняется при создании объекта из класса. Используется для инициализации состояния объекта при его создании

48 \_\_\_ в C# - это член, представляющий именованное значение или атрибут объекта. Используется для инкапсуляции данных объекта и обеспечивает возможность контролируемого доступа и изменения данных

49 \_\_\_ в C# может быть объявлено с помощью следующего синтаксиса: «public event EventHandler EventName;».

50 Ключевое слово \_\_\_ в C# используется для обозначения отсутствия значения или нулевой ссылки. Оно часто используется для инициализации переменных или объектов, которым еще не присвоено значение

51 \_\_\_ метод в C# - это метод без имени, который может быть определен и передан в качестве аргумента другому методу. Метод используются для создания встроенных экземпляров делегатов и упрощают код, уменьшая необходимость в отдельных именованных методах

52 \_\_\_ в C# - это тип, определяющий ссылку на метод. Используются для передачи методов в качестве аргументов другим методам, что позволяет динамически вызывать методы. Используется для обработки событий и обратных вызовов, а также для других целей

53 Аспект межплатформенного программирования, связанный с общими интерфейсами

54 \_\_\_ класс в C# - это класс, который разделен на несколько файлов, но при компиляции рассматривается как один класс. Используется для разделения определения класса на несколько файлов, что облегчает работу над большими классами и упорядочивает код

55 \_\_\_ метода в C# - это метод, который имеет то же имя, что и другой метод, но имеет другое количество или типы параметров. Используется для обеспечения нескольких реализаций метода, которые могут быть вызваны на основе аргументов, переданных методу

56 \_\_\_ в C# - это структура данных, которая содержит набор элементов. Общие типы \_\_\_ в C# включают List, Dictionary, Queue, Stack и Array

57 \_\_\_ метод в C# - это метод, который может выполняться асинхронно, что позволяет выполнять длительные задачи без блокировки вызывающего потока. Метод используется для повышения производительности и отзывчивости приложений путем разгрузки трудоемких задач на отдельные потоки

58 \_\_\_ в C# - это тип значения, который может содержать члены данных и методы. Похожа на классы, но у есть несколько ключевых отличий, например, хранится в стеке, а не в куче, и является типоми значения, а не типоми ссылок. Это означает, что она хранится непосредственно в памяти, в то время как классы хранятся в куче и доступ к ним осуществляется через ссылки

59 \_\_\_ тип в C# - это тип, которому можно присвоить значение или установить его в null. Этот тип используется для представления значения, которое может отсутствовать

60 \_\_\_ классы обеспечивают переносимость интерфейсов между разными платформами в межплатформенной разработке

61 \_\_\_ структуры данных C# предпочтительны для эффективной обработки больших объемов данных в межплатформенных приложениях

62 Механизм, обеспечивающий безопасность данных в межплатформенных приложениях

63 методы автоматизированного тестирования, которые обеспечивают проверку совместимости межплатформенных приложений с различными устройствами

64 Термин означает "кросс-платформенная разработка"

65 под "MVP (Model-View-Presenter)" в архитектуре программного обеспечения подразумеваетс

66 Методология "Agile" включает в себя следующие принципы программного обеспечения

Сложные (3 уровень)

67 "паттерн Singleton" в объектно-ориентированном программировании представляет собой

68 Характеристики, которые определяют "реактивное программирование" в современных приложениях

69 "RESTful API" веб-разработке представляют собой

70 Паттерны проектирования, которые связаны с "архитектурой микросервисов"

**Карта учета тестовых заданий**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Компетенция | {{ компетенция }} | | | |
| Индикатор | {{ индикатор }} | | | |
| Дисциплина | Межплатформенное програмиирование | | | |
| Уровень освоения | Тестовые задания | | | Итого |
| Закрытого типа | | Открытого типа |
| Альтернативный выбор | Установление соответствия/ последовательности | На дополнение |
| 1.1.1 (20%) | 5 | 2 | 7 | 14 |
| 1.1.2 (70%) | 17 | 7 | 24 | 48 |
| 1.1.3 (10%) | 3 | 1 | 4 | 8 |
| Итого: | 25 шт. | 10 шт. | 35 шт. | 70 шт. |

**Критерии оценивания**

**Критерии оценивания тестовых заданий**

Критерии оценивания: правильное выполнение одного тестового задания оценивается 1 условным баллом, неправильное – 0 баллов.

Максимальная общая сумма баллов за все правильные ответы составляет наивысший балл – 100 баллов.

**Шкала оценивания результатов компьютерного тестирования обучающихся** (рекомендуемая)

|  |  |  |
| --- | --- | --- |
| Оценка | Процент верных ответов | Баллы |
| «удовлетворительно» | 70-79% | 61-75 баллов |
| «хорошо» | 80-90% | 76-90 баллов |
| «отлично» | 91-100% | 91-100 баллов |

**Ключи ответов**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№ тестовых заданий** | **Номер и вариант правильного ответа** |  |  | 36 | C\#, c\# |
| 1 | Б |  |  | 37 | var |
| 2 | А |  |  | 38 | dynamic |
| 3 | А |  |  | 39 | using |
| 4 | А |  |  | 40 | try catch, catch try |
| 5 | А |  |  | 41 | Массив |
| 6 | А |  |  | 42 | Cписок |
| 7 | А |  |  | 43 | наследования, наследование |
| 8 | А |  |  | 44 | интерфейсом, интерфейс |
| 9 | А |  |  | 45 | делегатом, делегат, delegate |
| 10 | А |  |  | 46 | Перечисление |
| 11 | А |  |  | 47 | Конструктор |
| 12 | А |  |  | 48 | Свойство |
| 13 | А |  |  | 49 | Событие |
| 14 | А |  |  | 50 | null |
| 15 | А |  |  | 51 | Анонимный |
| 16 | А |  |  | 52 | Делегат |
| 17 | А |  |  | 53 | Полиморфизм |
| 18 | А |  |  | 54 | Частичный |
| 19 | А |  |  | 55 | Перегрузка |
| 20 | А |  |  | 56 | Коллекция |
| 21 | А |  |  | 57 | Асинхронный |
| 22 | А |  |  | 58 | Структура |
| 23 | А |  |  | 59 | Нулевой |
| 24 | А |  |  | 60 | Абстрактные, Абстрактный |
| 25 | А |  |  | 61 | Коллекции |
| 26 | 1-А, 2-Б, 3-В, 4-Г |  |  | 62 | Шифрование |
| 27 | 1-А, 2-Б, 3-В, 4-Г |  |  | 63 | Совместимость |
| 28 | 1-А, 2-Б, 3-В |  |  | 64 | Универсальность |
| 29 | 1-А, 2-Б, 3-В |  |  | 65 | Представление |
| 30 | 1-А, 2-Б, 3-В |  |  | 66 | Итерации |
| 31 | 1-А, 2-Б, 3-В |  |  | 67 | Единственный экземпляр |
| 32 | 1-А, 2-Б, 3-В |  |  | 68 | Потоки данных |
| 33 | 1-А, 2-Б, 3-В |  |  | 69 | Маршруты |
| 34 | 1-А, 2-Б, 3-В |  |  | 70 | Декомпозиция |
| 35 | 1-А, 2-Б, 3-В |  |  |  |  |