Энгельсский технологический институт (филиал)
федерального государственного бюджетного образовательного учреждения
высшего образования
«Саратовский государственный технический университет имени Гагарина Ю.А.»

Кафедра «Естественные и математические науки»

Оценочные материалы по дисциплине
Б.1.1.28 «Объектно-ориентированное программирование»

направления подготовки
09.03.04 «Программная инженерия»

профиль

«Управление разработкой программных проектов»

Энгельс 2024

1. Перечень компетенций и уровни их сформированности подисциплинам (модулям), практикам в процессе освоения ОПОП ВО

В процессе освоения образовательной программы у обучающегося в ходе изучения дисциплины «Объектно-ориентированное программирование» должны сформироваться компетенции: ОПК-2.

Критерии определения сформированности компетенций на различных уровнях их формирования

Индекс компетенции	Содержание компетенции
ОПК-2	Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности

Код и наименование индикатора достижениякомпетенции	Виды занятий для формирования компетенции	Оценочные средствадля оценки уровня сформированности компетенции
ИД- 5 $_{ОПК-2}$ Понимает принципы объектно-ориентированной парадигмы программирования и способен использовать ее при решении задач профессиональной деятельности.	Лекции, практические занятия, самостоятельная работа	Вопросы для устного опроса Практические задания Вопросы к зачету

Уровни освоения компетенции

Уровень освоения компетенции	Критерии оценивания
Продвинутый (отлично)	Знает: в полном объеме принципы объектно-ориентированной парадигмы программирования. Умеет: в полном объеме использовать объектно-ориентированный подход при разработке программ для решения практических задач любой сложности. Владеет: способностью в полном объеме использовать объектно-ориентированную парадигму при решении задач профессиональной деятельности.
Повышенный (хорошо)	Знает: на хорошем уровне принципы объектно-ориентированной парадигмы программирования. Умеет: использовать объектно-ориентированный подход при разработке программ для решения практических задач средней сложности Владеет: способностью использовать объектно-ориентированную парадигму при решении задач средней сложности профессиональной деятельности.
Пороговый (базовый) (удовлетворительно)	Знает: основные понятия и определения объектно-ориентированной парадигмы программирования. Умеет: использовать объектно-ориентированный подход при разработке программ для решения легких практических задач. Владеет: способностью использовать объектно-ориентированную парадигму при решении легких задач профессиональной деятельности.

2. Методические, оценочные материалы и средства, определяющие процедуры оценивания сформированности компетенций (элементов компетенций) в процессе освоения ОПОП ВО

2.1. Оценочные средства для текущего контроля

Вопросы для устного опроса

Тема 1. Класс ввода-вывода Console. Исключительные ситуации

1. Как называется стандартный класс, предназначенный для работы с консолью? В какомпространстве имен он определен?
2. Дайте общую характеристику методов ввода-вывода данных класса Console.
3. В чем заключается отличие методов Write и WriteLine?
4. Каким образом можно несколько строк задать как один параметр метода WriteLine?
5. Каким образом организуется форматный вывод с помощью методов Write и WriteLine?
6. Каким образом задается параметр формата методов Write и WriteLine?
7. Какие спецификаторы формата Вы знаете?
8. Каким образом на экран можно вывести текущую дату и время?
9. Что такое пользовательский шаблон форматирования?
10. В чем заключается отличие методов Read и ReadLine?
11. Каким образом может осуществляться ввод числовых значений?
12. Какие свойства и методы классы кроме методы ввода-вывода Вы знаете?
13. Каким образом заданную строку можно вывести ровно в центре экрана?
14. Что такое исключительная ситуация? Приведите примеры типичных исключительных ситуаций.
15. Что происходит с программой, если возникла исключительная ситуация и она не обработана? Какие возможности программисту предоставляет механизм обработки исключительных ситуаций?
16. Назовите основное достоинство механизма обработки исключительных ситуаций.
17. Какие преимущества дает механизм обработки исключительных ситуаций при работе сфункциями?
18. Что можно определить как исключительную ситуацию в программе на языке C#?
19. Назовите два основных способа генерации исключений в программе.
20. Какие стандартные классы исключений Вы знаете?
21. Опишите синтаксис конструкции try.
22. Что такое контролируемый блок? Какие операторы в него могут входить?
23. Что такое обработчики исключений catch?
24. Что такое завершающий блок finally? Какие операторы в него могут входить?
25. Опишите синтаксис конструкции throw. Что может являться параметром этой конструкции?
26. Какие свойства класса Exception Вы знаете?
27. Где может быть описана конструкция throw? Где конструкция throw чаще описывается напрактике?
28. Назовите три формы записи обработчика исключений. В каких случаях применяется каждая изних?
29. Сформулируйте правило расположения блоков обработчиков исключений.
30. Что такое процесс распространения исключений?
31. Опишите общий механизм обработки исключений
32. Каким образом контролируется процессом генерации исключений, возникающих припереполнении?
33. Какие операторы при написании программ следует заключать в блок try? Сколько обработчиковисключений рекомендуется создавать?
34. В каких фрагментах программы рекомендуется генерировать исключение?

Тема 2. Массивы в языке C#. Символы и строки в языке C#

1. Что такое массив, размерность массива, индексы массива?

2. На какие виды по способу выделения памяти делятся массивы? К какому из этих видовотносятся массивы в языке C#?
3. К каким типам данных относятся массивы в C# - ссылочным или значимым?
4. Какие типы массивов определены в языке C#?
5. Наследниками какого класса являются все массивы в языке C#?
6. Что может являться элементами массива?
7. Как описываются одномерные массивы в языке C#?
8. Каким образом можно инициализировать одномерный массив при описании?
9. Какие значения будут присвоены элементам массива, если он не инициализирован приописании?
10. Каким образом происходит обращение к элементу одномерного массива? Какое исключениегенерируется, если значение индекса выходит за границы массива?
11. Каким образом можно присвоить элементу массива случайное число?
12. Пусть описаны два массива int []a= {1,2},b={3,4,5}. Что произойдет при выполнении операцииa=b?
13. С помощью каких конструкций цикла можно работать с массивами в языке C#?
14. В чем заключается особенность конструкции цикла foreach? В каких случаях ее целесообразноиспользовать?
15. Каким образом можно безопасно обратиться к последнему элементу одномерного массива?
16. Каким образом в языке C# описываются двумерные массивы? Каким образом происходитобращение к элементу двумерного массива?
17. Каким образом может быть инициализирован двумерный массив?
18. Каким образом можно безопасно обратиться к последнему элементу одномерного массива?
19. Каким образом можно скопировать элементы одного массива в другой?
20. Какие свойства класса Array Вы знаете?
21. Какие методы класса Array Вы знаете?
22. Каким образом можно безопасно обратиться к последнему элементу двумерного массива?
23. Каким образом в языке C# описываются ступенчатые массивы?
24. Каким образом может быть инициализирован ступенчатый массив?
25. Как, используя вложенные циклы foreach, можно вывести на экран все элементы ступенчатогомассива?
26. Какие способы хранения текстовой информации используются в языке C#
27. Что представляет из себя тип данных char?
28. Какие методы класса Char Вы знаете?
29. Каким образом можно инициализировать массив символов?
30. Где на практике чаще всего используются массивы символов?
31. Что представляет из себя тип данных string? К каким типам данных он относится? Какиесуществуют ограничения при работе с этим типом данных?
32. Какие способы инициализации строк типа string Вы знаете?
33. Какие операции класса String Вы знаете?
34. Какие методы класса String Вы знаете?
35. Для чего используется класс StringBuilder? В каком пространстве имен он определен?
36. Какие операции класса StringBuilder Вы знаете?
37. Каким образом, на практике органзуется работа со строками типа StringBuilder.

Тема 3. Описание классов в C#.

1. Что такое класс в ООП? Какой принцип ООП реализует механизм классов?
2. Что такое класс в языке C#?
3. Каким образом описываются классы в C? Где может быть описан класс?
4. Что определяют спецификаторы доступа класса? Какие спецификаторы доступы Вы знаете?

5. На какие группы делятся все элементы класса? Какие элементы входят в каждую из групп.
6. Каким образом синтаксически отличаются поля класса от методов класса?
7. Чем отличается работа с полями и методами класса от работы с обычными переменными и функциями.
8. Как описываются поля и константы класса. Чем они отличаются при использовании?
9. Какие спецификаторы доступа для полей и констант класса Вы знаете? Какие из них рекомендуется использовать на практике?
10. Что такое статические поля? Для чего они используются? Чем статические поля отличаются от констант?
11. Для чего используется спецификатор readonly?
12. Что называются интерфейсом класса?
13. Что такое метод класса? Что такое заголовок метода и тело метода?
14. Что такое тип метода? Что такое фактические и формальные параметры метода.
15. Чем методы отличаются от обычных функций? Чем статический метод отличается от нестатического?
16. Что такое this? Для чего this применяется в явном виде?
17. Опишите действия, происходящие при вызове метода.
18. Чем отличается передача параметра в метод по значению и по ссылке.
19. Какие четыре типа параметров методов существуют в C#. Чем они отличаются?
20. Что такое перегруженные методы?
21. Что такое конструктор класса?
22. Перечислите особенности описания конструкторов.

Тема 4. Методы классов. Наследование.

1. Что такое деструктор класса?
2. Что такое сборщик мусора? Как он работает? Как его работа связана с наличием деструктора?
3. Для чего используется деструктор в языке C#?
4. Как описывается деструктор в языке C#?
5. Что такое свойства класса? С какой целью они используются?
6. Какие возможны стратегии доступа к полям класса?
7. Приведите синтаксис описания свойства класса? Зачем используется секции set и get?
8. Каким образом в программе происходит обращение к свойствам объекта?
9. Что такое индексаторы? Для чего они предназначены?
10. Приведите синтаксис описания индексаторов.
11. Что обычно является параметрами индексаторов?
12. Для чего используется перегрузка операций? Для каких классов чаще всего на практике перегружают операции.
13. Какие виды операций можно перегружать в языке C#?
14. Приведите синтаксис перегруженных операций в языке C#.
15. Опишите правила перегрузки операций в языке C#.
16. Какие унарные операции можно перегружать в языке C#, какие у них должны быть параметры икакие значения они должны возвращать?
17. Какие бинарные операции можно перегружать в языке C#, какие у них должны быть параметры икакие значения они должны возвращать?
18. Назовите принципы перегрузки операций отношения.
19. Приведите синтаксис перегруженных операций явного и неявного преобразования типа в языкеC#.
20. В каких случаях осуществляется явное, а каких случаях неявное преобразование типа.
21. Дайте характеристику методу Main в программе на языке C#.
22. Назовите два способа описания метода Main.
23. Каким может быть тип метода Main, как обычно интерпретируется возвращаемое значение?
24. Каким образом задаются фактические параметры для метода Main.
25. В чем заключается идея наследования в ООП?
26. Сколько потомков и предков может иметь класс в C#.

27. Приведите синтаксис класса-потомка.
28. Если при описании класса имя предка не указано, что это означает?
29. Назовите основное правило наследования полей и методов класса.
30. Что означает термин «скрытие» поля? Каким образом оно осуществляется? Каким образом можно получить доступ к срытому полю базового класса?
31. Какие варианты возможны при описании в производном классе метода с таким же именем что и в базовом?
32. Перечислите правила наследования конструкторов.
33. Что происходит, если в производном классе не определен ни один конструктор.
34. Как явным образом из конструктора производного класса вызвать конструктор базового класса?
35. Что происходит, если в конструкторе производного класса не вызван явным образом конструктор базового класса?
36. Каким образом вызываются конструкторы в случае иерархии классов?
37. Объекты каких классов могут быть присвоены объекту базового класса?
38. Что такое механизм раннего (статического) связывания?
39. Что такое механизм позднего (динамического) связывания?
40. Что такое виртуальный метод?
41. За счет чего реализуется механизм позднего связывания?
42. Как описываются и переопределяются виртуальные методы?
43. Какие методы рекомендуется делать виртуальными?
44. Какой принцип ООП реализуется с помощью виртуальных методов?
45. В каких случаях, во время компиляции программы может быть неизвестно, с объектом какого класса работает программа.
46. Для чего предназначены абстрактные классы? Можно ли создавать объекты абстрактных классов?
47. Что такое абстрактный класс?
48. Что такое абстрактный метод?
49. Что такое бесплодные классы? Как они описываются?
50. Для чего предназначены бесплодные классы?
51. Что такое отношение включения между классами?
52. Что означает, если метода описан со спецификатором sealed?
53. Для каких методов может быть указан спецификатор sealed?
54. Какие методы класса object Вы знаете? Каким образом они могут быть использованы в классах, создаваемых программистом?
55. В каких случаях класс object используется непосредственно?

Тема 5. Интерфейсы и структурные типы. Работа с файлами

1. Что такое тип данных «интерфейс» в языке C#?
2. Чем интерфейсы отличаются от абстрактных классов?
3. Укажите синтаксис описания интерфейса
4. В чем заключается назначение интерфейсов?
5. Перечислите правила описания интерфейсов.
6. Где и как происходит реализация интерфейса.
7. Какими способами можно обращаться к методам интерфейса
8. Для чего используется операция is. Укажите ее синтаксис.
9. Для чего используется операция as. Укажите ее синтаксис.
10. Каким образом используются стандартные интерфейсы .NET
11. Какой стандартный интерфейс используется для сравнения объектов. Укажите заголовок егометода.
12. Какие преимущества дает использование интерфейса IComparable?
13. Какой интерфейс используется, если необходимо сортировать объекты по разным критериям.Укажите заголовок его метода. Каким образом должен быть реализован данный интерфейс.
14. Опишите правила перегрузки операций отношения.
15. Что происходит при присваивании объектов? Что такое клонирование?
16. Чем отличается поверхностное клонирование от глубокого клонирования?
17. Каким образом реализуется поверхностное клонирование?
18. Каким образом реализуется глубокое клонирование?

19. Что такое файл?
20. Что такое чтение в файл, запись из файла?
21. Что такое поток данных?
22. Что такое буфер?
23. С помощью чего в C# осуществляется работа с потоками?
24. Перечислите основные классы для работы с потоками и их назначение
25. Опишите последовательность действий для работы с файлом в языке C#
26. Каким образом можно создать поток класса **FileStream?**
27. Какие методы используются для чтения и записи данных в файл?
28. Какой метод класса FileStream используются для закрытия файла, что происходит при его использовании?
29. Какую последовательность действий следует применить для чтения из текстового файла последовательности чисел.

Тема 6. Структуры данных, коллекции и классы-прототипы.

1. Что такое коллекция (контейнер)? Приведите примеры коллекций. От чего зависит выбор той или коллекции.
2. Назовите преимущества и недостатки использования коллекций.
3. Что такое массив? Какие операции эффективно выполнять с массивом?
4. Что такое список? Какие операции эффективно выполнять со списком?
5. Что такое стек?
6. Что такое очередь?
7. Что такое граф?
8. Что такое бинарное дерево?
9. Что такое хеш-таблица, хеш-функция?
10. Что такое множество и бинарное множество?
11. В каких пространствах имен в C# определено множество стандартных классов, реализующих работу с коллекциями. Чем отличаются классы, описанные в этих пространствах имен?
12. Что такое класс-прототип (generic)?

Практические задания

Задание 1. Вывод строки на экран с использованием esp-последовательности

Каждому студенту рекомендуется выполнить хотя бы одно из упражнений 1–10.

Используя esc–последовательности, выведите на экран следующий текст.

1. Это строка,
   иначе –“стринг”,
   иначе – “строковый литерал”
2. Это звуковой
   сигнал!

(Сопроводите вывод строки звуковым сигналом)

3. Это строка?
   “Да!”
4. “ – это двойные кавычки
   ‘ – это апостроф
5. Это “строка”?
   Это “строка”!
   (Используя символ возврата на шаг, во второй строке удалите восклицательный знак)
6. Это строка?
   Это строка!
   (Замените слово “Это” во второй строке словом “Да”, используя возврат каретки)
7. \\Это комментарий?
   //Нет, это комментарий
8. C:\Program Files\Far

D:\Program Files\Far

(Замените символ D на символ F используя возврат каретки)

9. Используя шестнадцатеричные коды символов, выведите на экран текст:
   Цена=
   =100$
10. Используя шестнадцатеричные коды символов в кодировке Unicode, выведите на экран текст:
   Цена=
   =100$

Задание 2. Массивы в C#. Символы и строки

Каждому студенту рекомендуется выполнить хотя бы одно из упражнений 1–12.

1. Написать и протестировать метод, находящий сумму элементов заданной целочисленной квадратной матрицы, распложённых на диагонали, «ортогональной» главной. Метод должен генерировать хотя бы одно исключение. Квадратная матрица должна генерироваться случайным образом и выводиться на экран в методе Main.
2. Написать и протестировать метод, находящий сумму элементов заданного целочисленного ступенчатого массива, расположенных на первой и последней позиции каждой строки. Метод должен генерировать хотя бы одно исключение. Ступенчатый массив должен генерироваться случайным образом и выводиться на экран в методе Main.
3. Написать и протестировать метод, находящий произведение элементов каждой строки заданной целочисленной прямоугольной матрицы и возвращающий массив этих произведений. Метод должен генерировать хотя бы одно исключение. Прямоугольная матрица должна генерироваться случайным образом и выводиться на экран в методе Main.
4. Написать и протестировать метод, находящий произведение элементов каждой строки заданного целочисленного ступенчатого массива и возвращающий массив этих произведений. Метод должен генерировать хотя бы одно исключение. Ступенчатый массив должен генерироваться случайным образом и выводиться на экран в методе Main.
5. Написать и протестировать метод, находящий максимальный элемент каждой строки заданной целочисленной прямоугольной матрицы и возвращающий массив этих максимальных элементов. Метод должен генерировать хотя бы одно исключение. Прямоугольная матрица должна генерироваться случайным образом и выводиться на экран в методе Main.
6. Написать и протестировать метод, который преобразует заданную целочисленную прямоугольную матрицу таким образом, чтобы на месте первой строки находилась вторая, на месте второй – третья, и т.д., а на месте последней – первая. Результатом работы метода является преобразования матрица. Метод должен генерировать хотя бы одно исключение. Прямоугольная матрица должна генерироваться случайным образом и выводиться на экран в методе Main.
7. Написать и протестировать метод, который в заданной целочисленной прямоугольной матрице ищет все числа, равные заданному числу и возвращает массив, содержащий позиции таких чисел. Метод должен генерировать хотя бы одно исключение. Прямоугольная матрица должна генерироваться случайным образом и выводиться на экран в методе Main.
8. Написать и протестировать метод, который в заданном целочисленном ступенчатом массиве ищет все числа, равные заданному числу и возвращает массив, содержащий позиции таких чисел. Метод должен генерировать хотя бы одно исключение. Ступенчатый массив должен генерироваться случайным образом и выводиться на экран в методе Main.
9. Написать и протестировать метод, который возвращает прямоугольную матрицу, транспонированную из заданной целочисленной матрицы. (Операция транспонирования заключается в том, что строки и столбцы в исходной матрице меняются местами). Метод должен генерировать хотя бы одно исключение. Прямоугольная матрица должна генерироваться случайным образом и выводиться на экран в методе Main.

10. Написать и протестировать метод, который возвращает ступенчатый массив, сгенерированный на основе заданного целочисленного ступенчатого массива следующим образом. Из каждой строки заданного массива удаляется первый и последний элементов. Если строка заданного массива имеет 2 и менее элемента, то строка нового массива содержит один элемент, равный -1. Метод должен генерировать хотя бы одно исключение. Ступенчатый массив должен генерироваться случайным образом положительными числами и выводиться на экран в методе Main.

11. Написать и протестировать метод, который возвращает ступенчатый массив, сгенерированный на основе заданного ступенчатого массива следующим образом. Из каждой строки заданного массива удаляются все элементы равные нулю. Если все строки заданного массива равны нулю, то строка нового массива содержит один элемент, равный -1. Метод должен генерировать хотя бы одно исключение. Ступенчатый массив должен генерироваться случайным образом из нулей и единиц и выводиться на экран в методе Main.

Задание 3. Синтаксис описания класса

1. Описать класс РАЦИОНАЛЬНОЕ ЧИСЛО (поля: ЧИСЛИТЕЛЬ, ЗНАМЕНАТЕЛЬ).
   Операция класса: сложение двух чисел (+)
   Методы класса: поиск наибольшего числа в массиве рациональных чисел (статический метод), сокращение рационального числа
   Функция демонстрационной программы: вычисление суммы всех рациональных чисел из заданного массива.
2. Описать класс ИСТОРИЧЕСКОЕ СОБЫТИЕ (поля: ЧИСЛО, МЕСЯЦ, ГОД, СОБЫТИЕ).
   Операция класса: вычисление интервала в днях, прошедшего между двумя заданными историческими событиями (-);
   Статический метод класса: поиск наиболее позднего события в массиве событий.
   Функция демонстрационной программы: поиск в массиве заданного события по его названию.
3. Описать класс ДАТА (поля) ЧИСЛО, МЕСЯЦ, ГОД.
   Операция класса: вычисление даты, на N дней вперед по заданной (+).
   Статический метод класса: поиск в массиве дат всех дат заданного года.
   Функция демонстрационной программы: удаление из массива заданной даты
4. Описать класс ДАТА (поля: ЧИСЛО, МЕСЯЦ, ГОД).
   Операция класса: увеличение даты на один день (++)
   Статический метод класса: по году и порядковому номеру дня в году возвращающий соответствующую дату.
   Дополнительные функции демонстрационной программы: поиск в массиве дат самой поздней.
5. Описать класс АЛГЕБРАИЧЕСКИЙ ПОЛИНОМ (поля: СТЕПЕНЬ, КОЭФФИЦИЕНТЫ).
   Операция класса: сложение двух заданных полиномов (+).
   Статический метод класса: вывод на экран полиномов с максимальной степенью из массива полиномов.
   Функция демонстрационной программы: удаление заданного полинома из массива.
6. Описать класс АВТОМАШИНА (поля: МАРКА (задается из фиксированного списка), ГОД ВЫПУСКА, НОМЕР, ФАМИЛИЯ ВЛАДЕЛЬЦА).
   Операция класса: вычисление разницы в годах выпуска между двумя машинами (-)
   Статический метод класса: сортировка массива машин по фамилии владельца.
   Функция демонстрационной программы: поиск в массиве всех машин с заданной маркой.
7. Описать класс СТУДЕНТ (поля ФИО, ЧИСЛО, МЕСЯЦ, ГОД РОЖДЕНИЯ).
   Операция класса: вычисление разницы в возрасте (в днях) для двух студентов (-).
   Статический метод класса: поиск в массиве всех студентов заданного года рождения.
   Функция демонстрационной программы: удаление студента с заданной фамилией из массива.
8. Описать класс ИСТОРИЧЕСКОЕ СОБЫТИЕ (поля ГОД, СОБЫТИЕ, УРОВЕНЬ (международный, всероссийский, местный)). Дополнительные функции демонстрационной программы:
   Операция класса: вычисление разницы в годах между двумя событиями (-)

Статический метод класса: сортировка массива событий по комбинации (год, событие);
Функция демонстрационной программы: поиск в массиве событий всех событий заданного уровня.

9. Описать класс СТУДЕНТ (поля: ФИО, ГОД РОЖДЕНИЯ, НОМЕР ГРУППЫ (задается из фиксированного списка)). Дополнительные функции демонстрационной программы:
   Операция класса: вычисление разницы в возрасте для двух студентов (-).
   Статический метод класса: удаление из массива студента с определенной ФИО;
   Функция демонстрационной программы: поиск в массиве всех студентов с заданным номером группы.
10. Описать класс ТОВАР (поля: НАИМЕНОВАНИЕ ТОВАРА, СТРАНА-производитель (задается из фиксированного списка), ОБЪЕМ ПАРТИИ (в штуках)).
    Дополнительные Операция класса: увеличение объема партии товара на n%(*)
    Статический метод класса: сортировка массива товаров по наименованию;
    Функция демонстрационной программы: поиск всех товаров, импортируемых заданной страной.
11. Описать класс УЧЕНИК (поля: ФИО, ГОД ОБУЧЕНИЯ, НАЗВАНИЯ КЛАССА (БУКВА а-д)).
    Операция класса: перевод ученика в следующий класс (++)
    Статический метод класса: сортировка массива учеников по паре (год обучения, название класса);
    Функция демонстрационной программы: удаление ученика с заданной ФИО из массива.
12. Описать класс УЧЕНИК (поля: ФИО, ГОД ОБУЧЕНИЯ, НАЗВАНИЯ КЛАССА (БУКВА а-д)), ИТОГОВАЯ ОЦЕНКА)
    Операция класса: увеличение итоговой оценки (++)
    Статический метод класса: сортировка массива учеников по фамилии;
    Функция демонстрационной программы: поиск всех учеников с отличной итоговой оценкой

Задание 4 Наследование

1. Описать базовый класс CStr - строка. Обязательные поля класса CStr: поле для хранения символов строки, значение типа byte хранит длину строки в байтах. Обязательные методы должны выполнять следующие действия: конструктор без параметров; конструктор, принимающий в качестве параметра строку типа string, конструктор, принимающий в качестве параметра символ; свойства; получение длины строки; очистка строки (сделать строку пустой). Переопределить следующие операции: сложение (+) — конкатенация строк; операция (==) - проверка на равенство. Описать производный от CStr класс CDstr - десятичная строка. Строки данного класса могут содержать только символы десятичных цифр и символы - и +, задающие знак числа. Символы - или + могут находиться только в первой позиции числа, причем символ + может отсутствовать, в этом случае число считается положительным. Если в составе инициализирующей строки будут встречены любые символы, отличные от допустимых, десятичная строка принимает нулевое значение. Содержимое данной строки рассматривается как десятичное число. Класс CDStr содержит следующие методы: конструктор без параметров; конструктор, принимающий в качестве параметра строку типа string; свойства; метод, преобразующий данную строку в целое число. Переопределить следующие операции: вычитание (-) — арифметическая разность строк; операция >— проверка на больше (по значению); операция < — проверка на меньше (по значению). Написать демонстрационную программу.

2. Описать базовый класс CStr – строка (см. вариант 1). Описать производный от CStr класс CSStr – шестнадцатеричная строка. Строки данного класса могут содержать только шестнадцатиричные символы. Если в составе инициализирующей строки будут встречены любые символы, отличные от допустимых, строка принимает нулевое значение. Содержимое данной строки рассматривается как знаковое шестнадцатеричное число.. Класс CSStr содержит следующие методы: конструктор без параметров; конструктор, принимающий в качестве параметра строку типа string; свойства, метод, преобразующий данную строку в целое десятичное число, метод изменение знака на противоположный (перевод числа в дополнительный код). Переопределить следующие операции: сложение (+) — арифметическая сумма строк; операция (==) — проверка на равенство. Написать демонстрационную программу.

3. Описать базовый класс CStr – строка (см. вариант 1). Описать производный от CStr класс CStr_ID – строка - идентификатор. Строки данного класса строятся по правилам записи идентификаторов в

языке С и могут включать в себя только те символы, которые могут входить в состав С-идентификаторов. Если исходные данные противоречат правилам записи идентификатора, то создается пустая строка-идентификатор. Класс CStr_ID содержит следующие методы: конструктор без параметров; конструктор, принимающий в качестве параметра строку типа string; конструктор, принимающий в качестве параметра символ; свойства. Переопределить операцию больше (>) - проверка на больше. (Строка считается больше другой, если код символа первой строки в *i*-й позиции, *i* изменяется от 0 до *n*-1, где *n* — длина более короткой строки, больше кода символа в той же позиции во второй строке, длины строк могут не совпадать), операцию меньше (<) - проверка на меньше. Написать демонстрационную программу.

4. Описать базовый класс CStr – строка (см. вариант 1). Описать производный от CStr класс CStr_C – строка – комментарий. Строки данного класса строятся по правилам записи комментариев в С++. Если исходные данные противоречат правилам записи идентификатора, то создается пустая строка-идентификатор. Класс CStr_C содержит следующие методы: конструктор без параметров; конструктор, принимающий в качестве параметра строку типа string; свойства; перевод всех символов строки в верхний регистр; перевод всех символов строки в нижний регистр; поиск первого вхождения символа в строку; Переопределить следующие операции: сложение (+) - операция конкатенации строк; вычитание (-) — из строки (первый операнд) удаляются все символы, входящие в строку, — второй операнд, при этом может получиться пустая строка. Написать демонстрационную программу.
5. Описать базовый класс CStr – строка (см. вариант 1). Описать производный от CStr класс CComplex– комплексное число. Строки данного класса состоят из двух полей, разделенных символом *i*. Каждое из полей может содержать только символы десятичных цифр и символы + и -, задающие знак числа. Символы + и – могут находиться только в первой позиции числа, причем символ + может отсутствовать. Если исходные данные противоречат правилам записи комплексного числа, то комплексное число принимает нулевое значение. Класс CStr_ID содержит следующие методы: конструктор без параметров; конструктор, принимающий в качестве параметра строку типа string, свойства. Переопределить следующие операции: операция (==) – проверка на равенство; умножение (*). Написать демонстрационную программу.
6. Создать класс CPoint — точка. На его основе создать классы CcoloredPoint – цветная точка и Cline - линия. На основе класса CLine создать класс CcoloredLine – цветная линия. Все классы должны иметь методы для установки и получения значений всех координат. Классы CcoloredPoint и CcoloredLine должны иметь обязательные поля – 2 точки и имя, обязательные методы для изменения цвета и получения текущего цвета. Классы CLine и CcoloredLine должны иметь конструкторы без параметров, конструктор, принимающий в качестве параметров 2 точки (и цвет), свойства. Определить методы, проверяющие, являются ли линии параллельные, перпендикулярными. Переопределить операцию сравнения (==) – проверка на совпадение линий. Написать демонстрационную программу.
7. Описать класс Cpoint - точка. Описать класс прямоугольник CRectangle. Обязательные поля класса CRectangle: - имя прямоугольника, 4 объекта класса CPoint, 4 поля типа double – стороны прямоугольника. Обязательные методы должны выполнять следующие действия: конструктор без параметров (нулевого прямоугольника); конструктор, принимающий в качестве параметра 4 точки, свойства для каждого поля; метод получения площади, метод получения периметра. Описать производный класс CQuadrate – квадрат. Обязательные методы должны выполнять следующие действия: конструктор без параметров (нулевого квадрата); конструктор, принимающий в качестве параметра 4 точки, свойства; метод получения площади, метод получения периметра. Переопределить операцию (==) – сравнение по площади. Написать демонстрационную программу.
8. Описать класс CPoint - точка. Описать класс прямоугольник CRectangle (см. вариант 7). Описать производный класс CTrapezoid – трапеция. Обязательные методы должны выполнять следующие действия: конструктор без параметров (нулевого квадрата); конструктор, принимающий в качестве параметра 4 точки, свойства, метод получения площади, метод получения периметра, метод, проверяющий, является ли трапеция равнобедренной. Для классов CRectangle и CTrapezoid

определить метод, проверяющий две фигуры на пересечение. Написать демонстрационную программу.

9. Описать класс CPoint - точка. Описать класс четырехугольник CTetragon. Обязательные поля класса CTetragon: имя прямоугольника, 4 объекта класса CPoint, 4 поля типа double – стороны четырех угольника. Обязательные методы должны выполнять следующие действия: конструктор без параметров (нулевого прямоугольника); конструктор, принимающий в качестве параметра 4 точки, свойства, метод получения периметра. Переопределить операцию операцию (<) – сравнение по длине периметра. Описать производный класс CRectangle – прямоугольник. Обязательные методы должны выполнять следующие действия: конструктор без параметров (нулевого квадрата); конструктор, принимающий в качестве параметра 4 точки, свойства; метод получения периметра. Переопределить операцию (<) – сравнение по длине периметра. Для классов CTetragon и CRectangle определить метод, определяющий факт включения одной фигуры в другую. Написать демонстрационную программу.

10. Описать класс CPoint - точка. Описать класс четырехугольник CTetragon (см. задачу 9). Описать производный класс CParallelogram – параллелограмм. Обязательные методы должны выполнять следующие действия: конструктор без параметров (нулевого параллелограмма); конструктор, принимающий в качестве параметра 4 точки, свойства; метод получения площади, метод получения периметра. Переопределить операцию (<) – сравнения по площади. Для классов CTetragon и CParallelogram определить метод, перемещения на плоскости. Написать демонстрационную программу.

Задание 5. Интерфейсы

1. В классе РАЦИОНАЛЬНОЕ ЧИСЛО реализовать интерфейсы IComparable и IComparer переопределить операции сравнения для рациональных чисел.
   a) С помощью метода Array.Sort осуществить сортировку массива рациональных чисел: по значению числа, по знаменателю.
   b) Модифицировать метод поиск наибольшего числа в массиве рациональных чисел, осуществляя сравнение рациональных чисел с помощью операции >.
2. В классе ИСТОРИЧЕСКОЕ СОБЫТИЕ реализовать интерфейсы IComparable и IComparer и переопределить операции сравнения для исторических событий.
   a) С помощью метода Array.Sort осуществить сортировку массива рациональных чисел: по названию события, по полной дате.
   b) Модифицировать метод поиска наиболее позднего события в массиве событий, осуществляя сравнение событий с помощью операции >.
3. В классе ДАТА реализовать интерфейсы IComparable и и IComparer переопределить операции сравнения для дат, причем операцию == определить как сравнение по году.
   b) С помощью метода Array.Sort осуществить сортировку массива дат: по полной дате, по дате месяца.
   c) Модифицировать метод в массиве дат всех дат заданного года, осуществляя сравнение событий с помощью операции ==.

4.В классе ДАТА реализовать интерфейсы IComparable и IComparer и переопределить операции сравнения для дат.

a. С помощью метода Array.Sort осуществить сортировку массива дат: по полной дате, по году.
b. Модифицировать функцию демонстрационной программы - поиск в массиве дат самой поздней, осуществляя сравнение дат с помощью операции >.

5.В классе АЛГЕБРАИЧЕСКИЙ ПОЛИНОМ реализовать интерфейсы IComparable и IComparer и переопределить операции сравнения полиномов, причем операцию == как сравнение полиномов.

c. С помощью метода Array.Sort осуществить сортировку массива дат: по степени, по значению коэффициента нулевой степени.

d. Модифицировать функцию демонстрационной программы - удаление заданного полинома из массива, осуществляя в ней сравнение с помощью операции ==.

6.В классе АВТОМАШИНА реализовать интерфейсы IComparable и IComparer переопределить операции сравнения машин, причем операцию == как сравнение по Марке.

e. С помощью метода Array.Sort осуществить сортировку массива машин: по году выпуска, по марке.
f. Модифицировать функцию демонстрационной программы - поиск в массиве всех машин с заданной маркой, осуществляя в нем сравнение с помощью операции ==.

7.В классе СТУДЕНТ реализовать интерфейсы IComparable и IComparer и переопределить операции сравнения студентов, причем операцию == как сравнение студентов по году рождения.

g. С помощью метода Array.Sort осуществить сортировку массива студентов: по ФИО, по полной дате рождения.
h. Модифицировать метод класса - поиск в массиве всех студентов заданного года рождения, осуществляя в нем сравнение с помощью операции ==.

8.В классе ИСТОРИЧЕСКОЕ СОБЫТИЕ реализовать интерфейсы IComparable и IComparer и переопределить операции сравнения событий, причем операцию == как сравнение событий по уровню.

i. С помощью метода Array.Sort осуществить сортировку массива событий: по уровню, по комбинации (год, событие).
j. Модифицировать функцию демонстрационной программы - поиск в массиве событий всех событий заданного уровня, осуществляя в ней сравнение с помощью операции ==.

9.В классе СТУДЕНТ реализовать интерфейсы IComparable и IComparer и переопределить операции сравнения студентов, причем операцию == как сравнение студентов по ФИО.

k. С помощью метода Array.Sort осуществить сортировку массива студентов: по ФИО, по комбинации (Номер группы, ФИО).
l. Модифицировать метод класса - удаление из массива студента с определенной ФИО, осуществляя в нем сравнение с помощью операции ==.

10.В классе ТОВАР реализовать интерфейсы IComparable и IComparer и переопределить операции сравнения товаров, причем операцию == как сравнение товаров по стране-производителю.

m. С помощью метода Array.Sort осуществить сортировку массива товаров: по стране производителю, по комбинации (объем партии, наименование).
n. Модифицировать функцию демонстрационной программы - поиск всех товаров, импортируемых заданной страной, осуществляя в ней сравнение с помощью операции ==.

11.В классе УЧЕНИК реализовать интерфейсы IComparable и IComparer и переопределить операции сравнения учеников, причем операцию == как сравнение учеников по ФИО.

o. С помощью метода Array.Sort осуществить сортировку массива учеников: по ФИО, по комбинации (год обучения, название класса).
p. Модифицировать функцию демонстрационной программы - удаление ученика с заданной ФИО из массива, осуществляя в ней сравнение с помощью операции ==.

12.В классе УЧЕНИК реализовать интерфейсы IComparable и IComparer и переопределить операции сравнения учеников, причем операцию == как сравнение учеников по оценке.

q. С помощью метода Array.Sort осуществить сортировку массива учеников: по ФИО, по комбинации (итоговая оценка, год обучения, название класса).
r. Модифицировать функцию демонстрационной программы - поиск всех учеников с отличной итоговой оценкой, осуществляя в ней сравнение с помощью операции ==.

Задание 6. Работа с файлами

В задании к практической работе №3 добавить два пункта меню:

- начального формирование массива объектов на основе данных из текстового файла,
- запись массива объектов в текстовый файл.

2.2. Оценочные средства для промежуточного контроля

Вопросы к зачету

1. Основные принципы объектно-ориентированного программирования. Инкапсуляция, наследование, полиморфизм.
2. Преимущества объектно-ориентированного программирования.
3. Понятие класса и объекта класса.
4. Структура программы на языке C#. Понятия пространства имен. Директива using. Синтаксис описания простейшего класса и метода Main.
5. Классификация типов данных в языке C#: встроенные и определяемые программистом, простые и структурные, типы значения и ссылочные типы.
6. Встроенные типы языка C# . Преобразование типов.
7. Основные методы класса Console и их параметры: Write, WriteLine, Read, ReadLine. Особенности чтения данных с клавиатуры в C#.
8. Общий механизм обработки исключений. Преимущества исключений.
9. Синтаксис исключений. Перехват исключений. Операторы checked и unchecked.
10. Одномерные массивы в языке C#.
11. Двумерные массивы в языке C#.
12. Ступенчатые массивы в языке C#.
13. Класс System.Char.
14. Строки типа String.
15. Строки типа StringBuilder.
16. Синтаксис описания класса в C#. Типы элементов класса. Спецификаторы видимости элементов класса.
17. Синтаксис описания полей и констант класса в языке C#.
18. Синтаксис описания методов класса в языке C#. Перегрузка методов.
19. Ключевое слово this.
20. Конструкторы и деструкторы класса в языке C#.
21. Свойства класса в языке C#.
22. Метод Main в языке C#.
23. Индексаторы в языке C#.
24. Операции класса в языке C#.
25. Описание производного класса в языке C#. Правила наследования элементов класса.
26. Виртуальные методы.
27. Абстрактные классы. Бесплодные классы.
28. Класс object.
29. Синтаксис интерфейса в языке C#. Реализация интерфейса.
30. Работа с объектами через интерфейсы. Операции is, as. Интерфейсы и наследование.
31. Потоки байтов. Потоки символов. Двоичные потоки.
32. Коллекции в C#. Классы-прототипы.
33. Понятие и структура платформы MS.NET.
34. Структура Microsoft.NET Framework.
35. Среда Common Language Runtime (CLR). Преимущества платформы MS.Net

Оценивание результатов обучения в форме уровня сформированности элементов компетенций проводится путем контроля во время промежуточной аттестации в форме зачета:

а) оценка «зачтено» – часть компетенции сформированы на базовом уровне;

б) оценка «не зачтено» – часть компетенции не сформированы.

Критерии, на основе которых выставляются оценки при проведении текущего контроля и промежуточной аттестации приведены в табл. 1.

Оценки «Не зачтено» ставятся также в случаях, если обучающийся не приступал к

выполнению задания, а также при обнаружении следующих нарушений:

– списывание;

– плагиат;

– фальсификация данных и результатов работы.

Таблица 1 – Критерии выставления оценок при проведении текущего контроля и промежуточной аттестации

Шкала оценки	Оценка	Критерий выставления оценки
ухбалльная шкала	чтено	Обучающийся ответил на теоретические вопросы. Показал знания в рамках учебного материала. Выполнил практические задания. Показал удовлетворительные умения и владения навыками применения полученных знаний и умений при решении задач в рамках учебного материала
	зачтено	Обучающиеся при ответе на теоретические вопросы и при выполнении практических заданий продемонстрировал недостаточный уровень знаний и умений при решении задач в рамках учебного материала. При ответах на дополнительные вопросы было допущено множество неправильных ответов

2.3. Итоговая диагностическая работа по дисциплине

ЗАДАНИЯ ДЛЯ ДИАГНОСТИЧЕСКОЙ РАБОТЫ ПО ДИСЦИПЛИНЕ «ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ»

Номер задания	Правильный ответ *	Содержание вопроса	Компетенция	Код и наименование индикатора достижения компетенции
1.	b	Основное назначение ОПП a. -: разработка алгоритмически сложных программ b. +: разработка больших и надежных программных систем c. -: разработка программ с максимальным быстродействием d. -: разработка программ, максимально эффективно использующихпамять	ОПК-2	**ИД- 5 опк-2** Понимает принципы объектно-ориентированной парадигмы программирования и способен использовать ее при решении задач профессиональной деятельности.

2.	d	Какое из следующих определений переменной в языке C# содержитсинтаксическую ошибку
		a. -int i;
		b. -int i=10;
		c. -int i=new int();
		d. int i=new int (10);
			ОПК-2	ИД- 5 опк-2

3.	b	В чем заключается основное преимущество использования механизмаисключительных ситуаций a. исключения позволяют уменьшить использование конструкцийветвления b. исключения позволяют логически разделить вычислительныйпроцесс на две части - обнаружение аварийной ситуации и ее обработку c. -исключения позволяют сократить количество модулей вмногомодульных программах d. -исключения позволяются сократить количество конструкторов вклассах	ОПК-2	ИД- 5 опк-2
4.	c	Массивы в языке C# делятся на a. одномерные и двумерные b. одномерные и многомерные c. одномерные, многомерные и массивы массивов d. -одномерные, двумерные и ступенчатые	ОПК-2	ИД- 5 опк-2
5.	b	Наиболее трудоемким при разработке является следующий разделтехнического задания a. назначение разработки b. требования к программе или программному изделию c. требования к программной документации d. стадии и этапы разработки; e. порядок контроля и приемки	ОПК-2	ИД- 5 опк-2

6.	d	Какое из нижеследующих описаний массива на языке C# содержитсинтаксическую ошибку
		a. -int[] a= {1,2,3};
		b. -int n = 3; int[] a= new int[n];
		c. -const int n = 3; int[] a= new int[n];
		d. const int n = 3; int[] a= new int[n]{1,2,3,4};
			ОПК-2	ИД- 5 ОПК-2

7.	a	Со спецификатором доступа private в классе на языке C# следуетописывать a. элементы класса, которые должны быть невидимы вне класса b. элементы класса, которые должны быть видимы толькопроизводным классам c. элементы класса, которые должны быть видимы вне класса d. элементы класса, которые должны быть видимы из даннойпрограммы	ОПК-2	ИД- 5 ОПК-2
8.	c	Ссылка this в языке C# это a. -ссылка на класс b. -ссылка на объект класса c. ссылка на объект, вызвавший метод d. ссылка на нестатический метод класса	ОПК-2	ИД- 5 ОПК-2
9.	d	Какое из утверждений о методе Main в языке C# верно. Метод Main a. должен иметь хотя бы один параметр b. может иметь один параметр любого типа c. может иметь произвольное количество параметров d. может иметь один параметр типа string[]	ОПК-2	ИД- 5 ОПК-2
10.	a	Какое из утверждений о наследовании в языке C# верно a. класс в C# может иметь произвольное количество потомков c. класс в C# может иметь произвольное количество классов-предков предком d. класса может быть только класс b. если имя предка при описании класса не указано, значат у негонет предков	ОПК-2	ИД- 5 ОПК-2
11.	наследование	Один из основных принципов ООП, заключающийся в возможностисоздания иерархии классов, в которой производные классы получаютэлементов своих предков, могут их изменять и добавлять новые, называется _____	ОПК-2	ИД- 5 ОПК-2
12.	полиморфизм	Один из основных принципов ООП, заключающийся в возможности использовать в различных классах иерархии одно имя для обозначенияразличных методов называют _____	ОПК-2	ИД- 5 ОПК-2
13.	инкапсуляция	Один из основных принципов ООП, заключающийся в объединении данных с функциями их обработки в сочетании со срытием ненужнойдля использования этих данных информации, называется_____	ОПК-2	ИД- 5 ОПК-2
14.	класс	В ООП тип данных, определяемым пользователем, который содержитописание данных и функций для работы с этими данным, называется _____	ОПК-2	ИД- 5 ОПК-2
15.	методы	Элементы класса в ООП обычно делятся на поля и _____	ОПК-2	ИД- 5 ОПК-2

16.	поле	Пусть описан
		классclass C
		{
		int a;
		int A1();
		public int A() {get
		{retutn a;public C()
		{…}
		~C()
		}
		
		a - это ... класса (ответ из одного слова)
			ОПК-2	ИД- 5 ОПК-2

17.	Конструктор	Пусть описан
		классClass C
		{
		int a;
		int A1();
		public int A() {get
		{retutn a;public C()
		{…}
		~C()
		}
		C() это ... класса
			ОПК-2	ИД- 5 ОПК-2

18.	деструктор	Пусть описан
		классClass C
		{
		int a;
		int A1();
		public int A() {get
		{retutn a;public C()
		{…}
		~C()
		}
		~C() это ... класса
			ОПК-2	ИД- 5 ОПК-2

19.	методы	Пусть описан
		классClass C
		{
		int a;
		int A1();
		public int A() {get {retutn a;
		public C() {…}
		~C()
		}
		
		A1() это ... класса (ответ из одного слова)
			ОПК-2	ИД- 5 ОПК-2

20.	Объект (экземпляр)	Пусть описан
		классClass C
		{
		int a;
		int A1();
		public int A() {get {retutn a;
		public C() {…}
		~C()
		}
		...
		C B=new C();
		
		B это ... класса
			ОПК-2	ИД- 5 ОПК-2