Программа курса «Языки программирования»
мехмат, 1 курс, направление «Фундаментальная информатика и информационные технологии»
2 семестр 2019–2020 уч. г.

Раздел 1. Регулярные выражения. Файлы. Абстрактные типы данных и классы

Регулярные выражения

1. Недостатки метода строк s.Split. Понятие регулярного выражения. Методы Regex.Match, Regex.IsMatch, Regex.Matches и примеры их использования. Аналогичные экземплярные методы и их отличие от статических. Классы Match и MatchCollection.
2. Квантификаторы. Примеры использования.
3. Классы символов. Примеры использования.
4. Директивы нулевой длины. Примеры использования.
5. Группы. Повторение элементов группы. Ссылки на группы. Примеры использования.
6. Замена с помощью регулярных выражений. Примеры использования.

Файлы

7. Определение файла. Преимущества и недостатки файлов. Классификация файлов по типу компонент и по способу доступа.
8. Основные операции при работе с закрытыми и открытыми файлами.
9. Основные пространства имён и классы .NET, связанные с файлами. Класс файлового потока FileStream, операции чтения и записи для FileStream. Режимы открытия файла.
10. Механизм обработки исключений. Оператор try – catch, алгоритм его работы. Оператор try – finally. Обработка исключений при работе с файлами – примеры.
11. Оператор using и его использование при открытии-закрытии файла.
12. Файловый указатель. Действия с файловым указателем. Цикл по FileStream. Буферизация файлов.
13. Текстовые адаптеры и работа с ними: чтение, запись, цикл по текстовому файлу. Вывод числовой информации в текстовые потоки.
14. Кодировки текстовых файлов. Методы класса File для создания текстовых файлов.
15. Двоичные адаптеры. Считывание-запись с помощью двоичных адаптеров. Цикл по двоичному файлу на чтение и на запись. Исключение EndOfStreamException. Возведение всех элементов в квадрат.
16. Текстовые файлы как последовательности строк. Примеры преобразования текстовых файлов. Парсер строк текстового файла, хранящего данные о персонах.

Введение в классы. Классы как абстрактные типы данных. Стандартные классы .NET

17. Классы и объекты. Модификаторы защиты доступа. Свойства в C#. Автоматически реализуемые свойства. Отличие класса от структуры. Представление в памяти.
18. Операции ?. и ?? Примеры использования.
19. Абстрактный тип данных (АТД). Класс как конкретный и как абстрактный тип данных. Принцип отделения интерфейса от реализации, его преимущества.
20. АТД Стек, его интерфейс. Вычисление значения выражения в польской инверсной записи с помощью стека.
21. АТД и класс Очередь, его интерфейс. Примеры использования очереди: алгоритм заливки области.

22. Стандартные классы коллекций .NET: Stack<T>, Queue<T>, List<T>, HashSet<T>, SortedSet<T>, Dictionary<K,V>, SortedDictionary<K,V>, LinkedList<T>. Коллекции как последовательности.
23. Стандартный класс словаря (ассоциативного массива) Dictionary<K,V>. Ключи и значения. Инициализация, цикл по словарю. Индексное свойство.
24. Реализация словаря на базе списка пар. Делегирование.
25. Исключение KeyNotFoundException.
26. Примеры использования: словарь количества встречаемости слов в файле.

Раздел 2. Динамические структуры данных и рекурсия

Динамические структуры данных

27. Статические и динамические структуры данных. Списки: линейные и циклические, односвязные и двусвязные. Класс узла списка.
28. Основные операции с линейными односвязными списками: вставка в начало, удаление из начала, вставка после текущего, удаление следующего, проход по списку, поиск. Недостатки линейного односвязного списка.
29. Класс двусвязного списка. Основные операции с линейными двусвязными списками: инициализация, вставка элемента в начало и конец, вставка элемента в середину перед и после данного, удаление элемента в начале, середине и конце списка, проход по списку.
30. Стандартные классы LinkedList<T>, LinkedListNode<T>, их основные методы и свойства.
31. Сравнение асимптотической сложности операций двусвязных списков и массивов.
32. Реализация стека на базе линейного односвязного списка.
33. Реализация очереди на базе линейного односвязного списка.

Рекурсия

34. Рекурсия, примеры. Праворекурсивные и леворекурсивные определения, примеры. Прямая и косвенная рекурсия. Грамматика языка программирования и формы Бэкуса-Наура. Рекурсивные подпрограммы. Прямая и косвенная рекурсия.
35. Примеры рекурсивного зацикливания. Простейшие примеры рекурсии. Глубина рекурсии, текущий уровень глубины рекурсии. Рекурсивный спуск и рекурсивный возврат.
36. Графическое изображение рекурсии (2 способа).
37. Примеры рекурсии: факториал числа, степень числа (2 способа, отличающиеся глубиной рекурсии), нахождение минимального элемента в массиве (2 способа), вывод линейного односвязного списка. Анализ глубины рекурсии в каждом случае.
38. Доказательство завершимости рекурсии.
39. Программный стек: определения, рисунок. Вызов подпрограммы на этапе выполнения. Запись активации, её содержимое. Что происходит при вызове подпрограммы и при выходе из подпрограммы. Пример использования программного стека при рекурсии. Переполнение программного стека.
40. Формы рекурсивных подпрограмм (5 шт.). Функция Аккермана.
41. Примеры плохого использования рекурсии: числа Фибоначчи. Каскадная рекурсия и дерево рекурсивных вызовов. Оптимизация рекурсивного алгоритма вычисления чисел Фибоначчи: метод с концевой рекурсией.
42. Примеры использования рекурсии: ханойские башни. Глубина рекурсии, количество рекурсивных вызовов.

43. Быстрая сортировка Хоара. Алгоритмы Partition и QuickSort. Оценка количества операций при быстрой сортировке в среднем. Сравнение с пузырьковой сортировкой. Быстрая сортировка в худшем случае.
44. ~~Функции-генераторы последовательностей. Оператор yield return. Генератор бесконечной последовательности.~~
45. Генерация всех перестановок. Глубина рекурсии и количество рекурсивных вызовов.
46. Генерация всех подмножеств. Глубина рекурсии и количество рекурсивных вызовов.

Деревья

47. Деревья, примеры. Основные определения: вершины и ребра, корень, листья, крона, глубина дерева.
48. Бинарные деревья. Рекурсивное определение бинарного дерева. Идеально сбалансированное и полное бинарное дерево.
49. Обходы деревьев: инфиксный, префиксный, постфиксный. Вывод элементов, полученных в результате обхода. Определение количества элементов в бинарном дереве, поиск элемента.
50. Создание идеально сбалансированного бинарного дерева.
51. Определение количества листьев в бинарном дереве.
52. Поиск элемента в бинарном дереве.
53. Нерекурсивный обход бинарного дерева в ширину с помощью очереди. Нерекурсивный обход бинарного дерева в глубину с помощью стека.
54. Связь деревьев и рекурсии.
55. Бинарные деревья поиска (БДП). Основные операции при работе с БДП: добавление, поиск.
56. Сортировка деревом. Оценка количества операций при добавлении и поиске в БДП, при сортировке деревом. Сортировка деревом в худшем случае.
57. Удаление из БДП (только алгоритм).
58. Реализация множества на базе бинарного дерева поиска.
59. Реализация ориентированного графа в виде набора инцидентных вершин. Сериализация графа в файл в формате json и десериализация. Остовное дерево графа. Алгоритм обхода графа в глубину с получением глубинного остовного дерева. Алгоритм обхода графа в ширину с получением остовного дерева в ширину.
60. ~~Бинарные деревья - алгоритм определения минимальной суммы от корня к листу. Отсечение заведомо неверных частичных решений.~~
61. ~~Алгоритм перебора с возвратом. Обход конем шахматной доски. Оптимизация обхода конем шахматной доски.~~

Раздел 3. Объектно-ориентированное программирование

Наследование. Исключения

62. Наследование, примеры. Диаграмма наследования на UML. Пример наследования Person – Student. Переопределяющие методы. Вызов унаследованного конструктора. Цели наследования. Наследование как расширение и как сужение. Иерархия наследования – примеры. Хранение потомка в памяти.
63. Принцип «Открыт-закрыт» и его роль при проектировании сложных систем. Учет будущих изменений. Пример: очередь с фильтрацией.
64. Наследование и включение. Пример: очередь с фильтрацией, реализованная включением; недостатки.
65. Когда следует использовать наследование, а когда – включение. Примеры.

66. Виды отношений между классами: ассоциация, агрегация, композиция, наследование, реализация интерфейсов. UML-диаграммы классов. Пример UML-диаграммы Персона-Преподаватель-Студент-Старшекурсник-Группа-Кафедра-Институт.
67. Наследование и выявление общего предка.
68. Модификатор доступа protected.
69. Принцип подстановки Барбары Лисков.
70. Класс System.Exception, его свойства. Иерархия наследования исключений .NET. Обработка исключений разных типов – примеры. Порядок записи обработчиков исключений. Создание собственных типов исключений. Обработка всех исключений. Генерация исключения.
71. Приведение типов в иерархии предок-потомок. DownCast и UpCast.
72. Операции is и as.

Полиморфизм и интерфейсы

73. Определение полиморфизма. Раннее и позднее связывание.
74. Позднее связывание и виртуальные методы. Переопределение виртуального метода. Полиморфные переменные, статический и динамический тип. Цепочка виртуальности и её разрыв. Алгоритм поиска в цепочке виртуальности.
75. Виртуальные методы как блоки для замены кода.
76. Полиморфные контейнеры. Обработка всех элементов вызовом виртуальных методов (на примере метода Work()). Полиморфные контейнеры и операции as-is.
77. Класс Object – неявный предок всех классов в NET. Метод GetType() и его сравнение с операцией is. Пример – родословная объекта. Полиморфные контейнеры и метод GetType().
78. Переопределение методов ToString(), Equals() и GetHashCode() в классе Person. Возможность определения HashSet<Person>.
79. Интерфейсы. Что может и что не может присутствовать в интерфейсе. Реализация интерфейсов классами.
80. Совместимость по присваиванию и операции is as для интерфейсов. Интерфейсы и полиморфизм.
81. Интерфейсы как роли.
82. Стандартные интерфейсы NET. Интерфейс сравнения IComparable<T> и его использование для создания SortedSet<Person>. Интерфейс сравнения IComparer<T> и его использование для сортировки списка персон.
83. Ограничения, налагаемые на параметры обобщений в методах и классах. Секция where, виды ограничений. Примеры: обобщенная функция поиска минимального элемента в массиве, обобщённый класс MySortedSet<T> на базе бинарного дерева поиска.