QA
מסע

אלכס גורבצ'וב

АЛЕКСЕЙ ГОРБАЧЁВ

JAVA

JAVA

Коллекции в Java

Для хранения наборов данных в Java предназначены массивы. Однако их не всегда удобно использовать, прежде всего потому, что они имеют фиксированную длину. Эту проблему в Java решают коллекции. Однако суть не только в гибких по размеру наборах объектов, но в и том, что классы коллекций реализуют различные алгоритмы и структуры данных, например, такие как стек, очередь, дерево и ряд других.

Классы коллекций располагаются в пакете java.util, поэтому перед применением коллекций следует подключить данный пакет. Хотя в Java существует множество коллекций, но все они образуют стройную и логичную систему.

Java

Коллекции в Java

Основные коллекции:

- Collection: базовый интерфейс для всех коллекций и других интерфейсов коллекций
- List: наследует интерфейс Collection и представляет функциональность простых списков
- Set: также расширяет интерфейс Collection и используется для хранения множеств уникальных объектов
- SortedSet: расширяет интерфейс Set для создания сортированных коллекций
- NavigableSet: расширяет интерфейс SortedSet для создания коллекций, в которых можно осуществлять поиск по соответствию
- Map: предназначен для созданий структур данных в виде словаря, где каждый элемент имеет определенный ключ и значение. В отличие от других интерфейсов коллекций не наследуется от интерфейса Collection

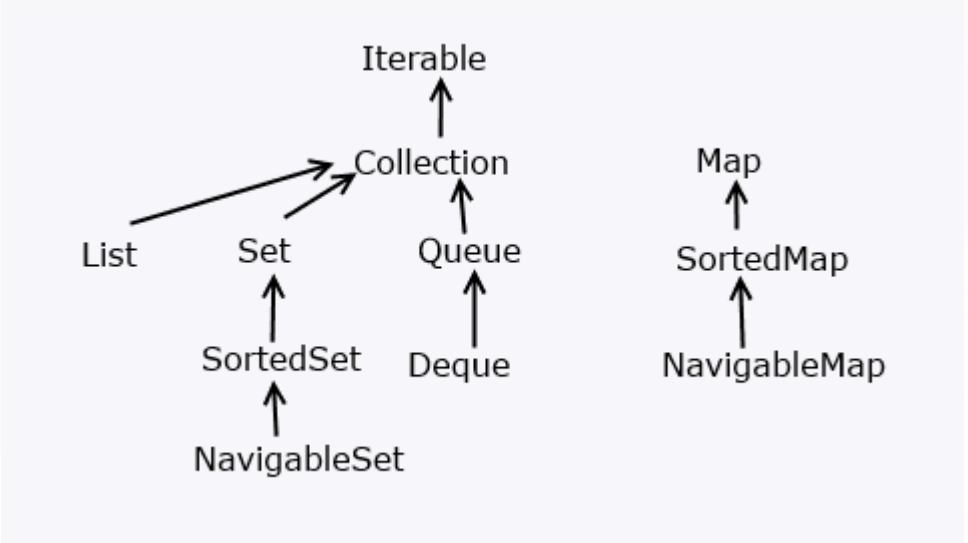
Коллекции в Java

С помощью применения вышеописанных интерфейсов и абстрактных классов в Java реализуется широкая палитра классов коллекций - списки, множества, очереди, отображения и другие, среди которых можно выделить следующие:

- ArrayList: простой список объектов
- LinkedList: представляет связанный список
- ArrayDeque: класс двунаправленной очереди, в которой мы можем произвести вставку и удаление как в начале коллекции, так и в ее конце
- HashSet: набор объектов или хеш-множество, где каждый элемент имеет ключ - уникальный хеш-код
- TreeSet: набор отсортированных объектов в виде дерева
- LinkedHashSet: связанное хеш-множество
- PriorityQueue: очередь приоритетов
- HashMap: структура данных в виде словаря, в котором каждый объект имеет уникальный ключ и некоторое значение
- TreeMap: структура данных в виде дерева, где каждый элемент имеет уникальный ключ и некоторое значение

Коллекции в Java

Схематично всю систему коллекций вкратце можно представить следующим образом:



Java

Коллекции в Java. Класс ArrayList и интерфейс List.

Класс ArrayList представляет обобщенную коллекцию, которая наследует свою функциональность от класса AbstractList и применяет интерфейс List.

Проще говоря, ArrayList представляет простой список, аналогичный массиву, за тем исключением, что количество элементов в нем не фиксировано.

Всегда используйте ArrayList вместо массивов.

Для того, чтоб можно было использовать ArrayList нужно добавить библиотеку:

```
import java.util.ArrayList;
```

Java

Коллекции в Java. Класс ArrayList и интерфейс List.

Array	ArrayList
Создание контейнера элементов	
String[ ] list = new String [10];	ArrayList<String> list = new ArrayList<>();
Получение количества элементов	
int n = list.length;	int n = list.size();
Взятие элемента из массива/коллекции	
String s = list[3];	String s = list.get(3);
Запись элемента в массив	
list[3] = s;	list.set(3, s);

Java

Коллекции в Java. Класс ArrayList и интерфейс List.

Methods of class ArrayList<E>

add(E element) – вставляет элемент element в конец списка.

add(int index, E element) – вставляет элемент element на позицию index.

remove(int index) – удаляет элемент по определенной позиции index.

remove(Object o) – удаляет первый найденный элемент o если он существует в списке.

clear() – удаляет все элементы списка.

get(int index) – возвращает элемент находящийся на позиции index.

set(int index, E element) – замещает элемент на позиции index элементом element.

size() – возвращает количество элементов в списке.

toArray() – возвращает массив, который содержит все элементы списка.

indexOf(Object o) – возвращает позицию элемента o в списке или -1 если элемента нет

contains(Object o) – возвращает true если список содержит элемент o

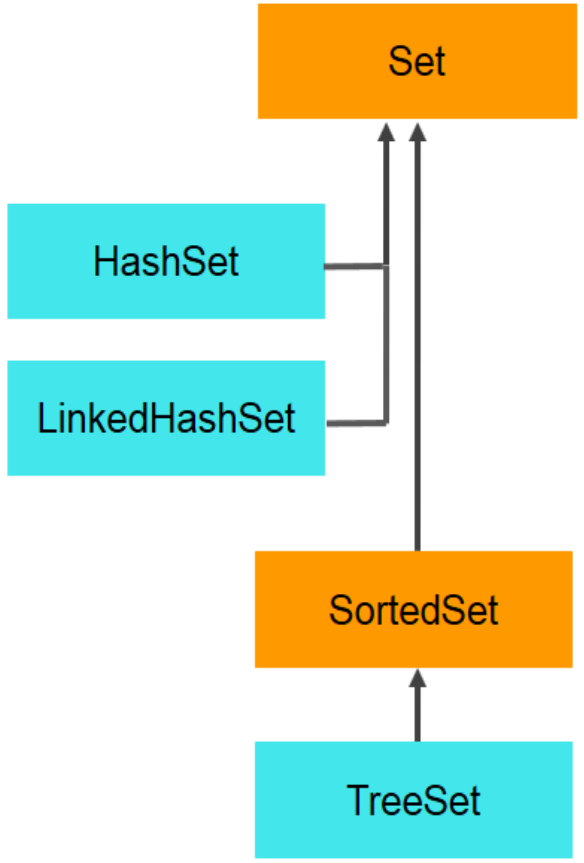
Java

Коллекции в Java. HashSet.

Множество (Set) - это такой же способ хранения данных, как массив или список. Но особенность множества в том, что оно может хранить только **уникальные** значения.

Среди HashSet, LinkedHashSet и TreeSet чаще всего используется HashSet.

HashSet хранит элементы в **произвольном** порядке, но зато быстро ищет. Подходит, если порядок Вам не важен, но важна скорость. Более того, для оптимизации поиска, HashSet будет хранить элементы так, как ему удобно.



Java

Коллекции в Java. HashSet.

Синтаксис:

```
HashSet<String> myHashSet = new HashSet<String>();
```

Для того, чтоб можно было использовать HashSet нужно добавить библиотеки:

```
import java.util.HashSet;
import java.util.Set;
```

Java

Задание.

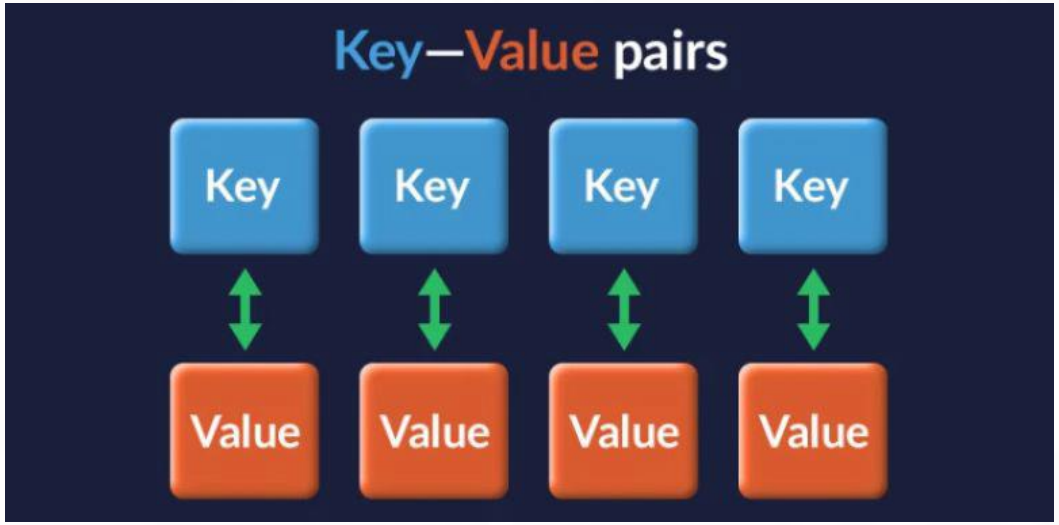
- Напишите программу, выдающую 6 целых чисел в диапазоне от 1 до 34. Используя nextInt(int n) и HashSet.
- Бонус – все 6 чисел разные
- Бонус – дополнительное число от 1 до 7

Java

Коллекции в Java. HashMap.

Интерфейс HashMap<K, V> представляет отображение или иначе говоря словарь, где каждый элемент представляет пару "ключ-значение".

При этом все ключи уникальные в рамках объекта Map. Такие коллекции облегчают поиск элемента, если нам известен ключ - уникальный идентификатор объекта (например: номер паспорта).



Java

Задание.

- Создайте класс Countries с помощью HashMap
  - Заполните его <String, String> (<City, Country>)
  - Создайте до 10-и пар.
  - Вывести на экран: Каждую страну.
  - Вывести на экран: Город - Страна.

- Создать 3 коллекции:
  - В первой 10 чисел
  - Во второй 10 имен (заполните их произвольными значениями)
  - Третья коллекция должна автоматически заполняться строками, которые содержат число из первой , коллекции, потом знак тире и строку из второй коллекции.
  - Вывести все значения третьей коллекции в цикле for-each.

Java

Методы (Функции) в Java.

Что такое метод?
Любой код на Java, который Вы откроете, будет состоять из методов. Можно сказать, это строительные "блоки", из которых состоит программа:

Java

Методы в Java.

Если Вы видите в программе какое-то слово, а затем круглые скобки - значит это метод: слово()

Например - это названия 4 методов:

- println()
- hasNextInt()
- getNumber()
- main()

Java

Методы в Java.

Зачем же они нужны?
Возьмём пример программы калькулятор.
В упрощенном виде программа-калькулятор должна выполнять 4 базовые операции: сложение, вычитание, умножение и деление. Поэтому, если бы мы писали программу-калькулятор, мы бы попросили пользователя:

1. Ввести 2 числа
2. Ввести операцию: "+" для сложения, "-" для вычитания, "*" для умножение и "/" для деления

Поэтому мы бы написали следующий код:

```
import java.util.Scanner;
public class Calculator {
    public static void main(String[] args) {
        double num1 = getNumber();
        double num2 = getNumber();
        char operation = getOperation();
        double result = calc(num1, num2, operation);
        System.out.println("Результат:" + result);
    }
}
```

Java

Методы в Java.

В каждой строчке выполняется метод:

```
class Calculator {

 main(){
    getNumber();
    getNumber();
    getOperation();
    calc(num1, num2, operation);
    System.out.println();
   }
}
```

```
class Calculator {

 main(){
    getNumber();   ← Просим ввести первое число
    getNumber();   ← Просим ввести второе число
    getOperation();   ← Просим ввести операцию
    calc(num1, num2, operation);   ← Считаем
    System.out.println();   ← Выводим результат
   }
}
```

Java

Методы в Java.

Все виды методов в Java можно поделить на две категории:

1. Стандартные, то есть написанные в стандартных библиотеках Java. Просто берешь нужный метод и используешь:
   - main()
   - println()
2. Пользовательские, то есть методы, которые Вы сами написали:
   - getNumber()
   - getOperation()
   - calc()

Java

Методы в Java.

Как строится метод?
Обычно мы используем: public static перед названием метода

Название метода

```
void myMethod(int x)
{
  System.out.println("You entered number " + x);
}
```

Я принимаю это

```
void myMethod(int x)
{
  System.out.println("You entered number " + x);
}
```

... и возвращаю это

```
void myMethod(int x)
{
  System.out.println("You entered number " + x);
}
```

Java

Задание.

- Необходимо написать метод, который бы возводил число в степень. И далее два числа, возведенные в степень, должны быть просуммированы, а результат выведен в консоль. Например:

$3^2 = 3*3 = 9$

$2^6 = 2*2*2*2*2*2 = 64$

$9 + 64 = 73$

Java

Задание.

- Напишите программу, которая принимает два числа - длину и ширину прямоугольника от пользователя. Затем программа должна вызывать метод, который принимает эти два числа и возвращает площадь прямоугольника.
- Напишите программу, которая запрашивает у пользователя целое неотрицательное число n и вычисляет его факториал. Факториал числа n обозначается как n! и равен произведению всех положительных целых чисел от 1 до n. Например, 5! = 5 * 4 * 3 * 2 * 1 = 120.

  Вам необходимо создать метод с именем factorial, который принимает целое число n в качестве аргумента и возвращает его факториал.

Thanks for your time ☺