アルゴリズムとデータ構造 2015 第６回　　年　　組　　番 氏名

1. 下記は，最も単純な**文字列探索**アルゴリズムである。このメソッドは，文字列 text の中に文字列 word が含まれるか先頭から調べ，最初に見つかった位置（0 以上）を返す。ただし，見つからなかった場合には -1 を返す。空欄を適切に埋めて実行し，動作を確認せよ。

```
public static int findString(String text, String word) {

    int tlen = text.length();
    int wlen = word.length();

    for (int t = 0; t <=          ; t++) {
        int w;
        for (w = 0; w <       ; w++) {

            if (text.charAt(      ) != word.charAt(    ))
                break;
        }
        if (w == wlen) return t;
    }

    return
}
```

2. 下記は商品コード（code）と価格（price）を持つ単純なクラス Item である。この**クラスの配列**の要素を価格が**高い順**にクイックソートで整列するメソッド sortByPrice を作成したい。空欄を適切に埋めよ。

```
public class Item {
    public int code;
    public int price;
}

public class ItemSorter {

    public static void sortByPrice(Item[] a, int left, int right) {

       if (left >= right) return;

       Item pivot =

       int l = left, r = right;
       while (true) {
           while (                  ) {

           }
           while (                  ) {

           }
           if (l >= r) break;

           Item t;

           l++; r--;
       }

       sortByPrice(a,         ,         );

       sortByPrice(a,         ,         );
    }
}
```

3. キーボードから文字列を 10 個配列に読み込み，Arrays.sort メソッドを用いて，アルファベット順に並び替えてから表示するプログラムを作成せよ。

4. 文字列の配列をアルファベット順の逆順に並び替えたい。しかし，通常の Arrays.sort は配列要素の compareTo メソッドを使うので，順序を指定することができない。そこで，以下の順序を指定できるバージョンを用いる。

```
static <T> void sort(T[] a, Comparator<? super T> c)
```

これはジェネリックメソッドであり、第 1 引数に String の配列を渡すと, T に String が当てはめられて，自動的に以下のようなメソッドができると思えばよい。

```
static void sort(String[] a, Comparator<? super String> c)
```

ここで問題なのは第 2 引数である。これは，ジェネリックインタフェース Comparator を実装し，クラス T（ここでは String）のインスタンス同士を比較できるクラスを表す。Comparator は，メソッドとして compare を持ち， それは T のインスタンス a，b を引数に取り，a が b に先行するなら負，a と b が同じ順位なら 0，a が b より後行するなら正の整数を返さなければならない。具体的には以下のようにする。return の部分はどう書けばよいか考えよ。

```
class StringReverseComparator implements Comparator<String> {
    public int compare(String a, String b) {

        return
    }
}
```

プログラムでは，このクラスのインスタンスを作成し，sort の第 2 引数に渡す。

```
Comparator<String> revcomp = new StringReverseComparator();
Arrays.sort(data, revcomp);
```

以上の説明を理解した上でプログラムを作成せよ。

※ 実は, 逆順にソートするための Comparator は既に用意されており, 以下のように書くことができる。

```
Arrays.sort(data, java.util.Collections.reverseOrder());
```

5. 整数を int 型の配列に読み込み, 配列の内容を大きい順に並べ替えてから表示するプログラムを作成せよ。実はこれは少し難しい。int はクラスでないので 4.の技法が使えない。Arrays.sort をした後，配列の内容をすべて逆順に入れ替えてから，最後に表示すればよい。

6. 2.で定義したクラス Item の配列を，Arrays.sort で code 順に並び替え，Arrays.binarySearch で探索ができるようにするためには、どのように修正すればよいか。

```
class Item implements Comparable<Item> {  // 比較可能であることを示すインタフェース

    public int code;
    public int price;

    @Override
    public boolean equals(Object obj) { /* プロ II の教科書の 14.2.6 を参考に実装 */ }

    @Override
    public int compareTo(Item item2) { /* code の小さい順になるように実装 */ }

    @Override
    public int hashCode() { return code; }  // equals をオーバーライドするときに必要

}
```