

1. 整数の配列 `data` から、値 `key` を探し出してその添字 (`index`: 配列の要素番号) を返す (戻す) メソッド (関数) を作成せよ。ただし、要素に値の重複はないものとし、もし `key` が `data` の中に含まれなかった場合には、`-1` を返すものとする。適当なクラス定義や `main` メソッドを補い、その動作を確認せよ。

```
public static int linearSearch(int key, int[] data) {

    int ret = -1;

    return ret;
}
```

2. 上記の線形探索において、配列 `data` の中に値 `key` がないという最悪の場合、`data` の要素と `key` の比較は何回行われるか n を用いて表せ (つまりループを何回繰り返して `if` 文を何回実行するか)。さらに、配列のサイズ n が k 倍になると、比較の回数は何倍になるか示せ。

3. 線形探索において配列 `data` の中に値 `key` が必ず 1 個含まれる場合、さまざまな `data` と `key` の組合せで実行すると、探索 1 回あたり `data` の要素と `key` の比較は平均して何回行われるか、以下の手順で求めよ。

- (a) まず、2.の結果を参考に平均の比較回数を直感的に予測し、 n を用いて表してみよ。
- (b) n 個のうちの 1 つの要素 `data[i]` に、`key` が入っている確率 P_i を示せ。
- (c) `data[0]` から始めて `data[i]` まで要素を順に `key` と比較する回数 N_i を示せ。
- (d) 平均の比較回数 \bar{N} は以下の計算で求められる (すなわち $\sum P_i N_i$ である)。計算結果を n の式で表せ。

$$\begin{aligned}
 & \text{key が data[0]にある確率 } P_0 \times \text{key が data[0]にある場合の比較回数 } N_0 \\
 & \text{key が data[1]にある確率 } P_1 \times \text{key が data[1]にある場合の比較回数 } N_1 \\
 & \quad \dots \\
 & +) \underline{\text{key が data[n-1]にある確率 } P_{n-1} \times \text{key が data[n-1]にある場合の比較回数 } N_{n-1}}
 \end{aligned}$$

4. 2.の値を最大 (最悪) 計算量, 3.の値を平均計算量という。それぞれを O 記法 (オーダー記法) で表せ。
 5. 下記のように、Object クラスを用いれば、多態性 (ポリモーフィズム) によって任意のクラスに対応する線形探索が定義可能である。プログラムとして完成させ、String や Integer クラスの配列で試してみよ。
- ```
public static int linearSearch(Object key, Object[] data)
```