Question 2

Please indicate the correct choice by returning the appropriate `char []` array from the corresponding method in the `ToughChoices` class!

For example, the following means the answer to the first MCQ is `a` and `c`, whereas the answer to the second question is `b`.

```
public static char q1() {
    return new char[] {'a', 'c'};
}

public static char q2() {
    return new char[] {'b'};
}
```

Multiple-Choice Questions

1. Linked lists are not suitable for the implementation of
    - a. Insertion sort
    - b. Linear search
    - c. Bubble sort
    - d. Binary search
    -
2. Where should you orient the front of a Stack if it is being implemented with a singly linked list with a head reference and optional tail reference so that all operations are constant time?
    - a. at the head
    - b. at the tail
    - c. either one
    - d. it cannot be done
3. Expand and simplify (if possible), then determine the big-Oh asymptotic complexity for

$$f(n) = 13 + 4\log n + \frac{18n^2 + n\log n}{2n}$$

    - a. $O(\log n)$
    - b. $O(n)$
    - c. $O(n \log n)$
    - d. other

4. Expand and simplify (if possible), then determine the big-Oh asymptotic complexity for

$$f(n) = n\log(n^2) + 3n(n+1)$$

   - a. $O(n^2)$
   - b. $O(n)$
   - c. $O(n \log n)$
   - d. other

5. Suppose we have the following array contents after the third pass of the outer loop of some quadratic sorting algorithm meant to put the array in ascending order: `[3, 5, 7, 4, 2, 9, 8, 10, 15, 20]`. Which sorting algorithm could be operating on this array?

   - a. bubble (up) sort
   - b. (min) selection sort
   - c. insertion sort
   - d. none of these

6. Consider these two methods that calculate the same function of `n`:

```java
public static double func1(int n) {
  if (n >= 1) {
      return 2 * func1(n - 1);
  }
  else return 1.0;
}
```

```java
public static double func2(int n) {
  double result = 1.0;
  for (int i = 1; i <= n; i++) {
    result = result * 2;
  }
  return result;
}
```

   Which of the following statements are true?

   - a. `func2` has $O(1)$ input space complexity
   - b. `func1` has $O(1)$ auxiliary space complexity
   - c. `func2` has $O(n)$ input space complexity
   - d. `func1` has $O(n)$ time complexity