

6. [Java] ?????? - list.toArray(new String[0]);

<https://school.programmers.co.kr/learn/courses/30/lessons/120897>



```
import java.util.*;

class Solution {
    public int[] solution(int n) {
        ArrayList<Integer> list = new ArrayList<>();
        for (int i = 0; i <= n; i++) {
            if(n%i==0) list.append(i);
        }
        int[] answer = list.toArray(new String[0]);
        return answer;
    }
}
```

실행 결과

```
/Solution.java:7: error: cannot find symbol
    if(n%i==0) list.append(i);
                    ^
    symbol:   method append(int)
    location: variable list of type ArrayList<Integer>
/Solution.java:9: error: incompatible types: inference variable T has incompatible bounds
    int[] answer = list.toArray(new String[0]);
                               ^
    lower bounds: int,Object
    lower bounds: String
    where T is a type-variable:
      T extends Object declared in method <T>toArray(T[])
2 errors
```

테스트 결과 (~~~)~

2개 중 0개 성공

? for (int i = 0; i <= n; i++)

- i = 0 일 때 n % 0 일 경우 ArithmeticException (0으로 나눌 수 없음) 발생
- i = 1 일 때 n % 1 일 경우 항상 0이므로 append() 호출되지 않음

? list.add(i)

- ArrayList의 add() 메서드는 append()와 동일함
- list.append(i) → list.add(i)

? int[] answer = list.toArray(new String[0]);

- List → Array로 변환하기 위해 toArray() 메서드를 사용함. 이때 String[]을 전달하면 String[]을 반환함
- String[]을 int[]로 변환하기 위해 for문을 사용함

import java.util.*;

```
import java.util.*;

class Solution {
    public int[] solution(int n) {
```

```

    ArrayList<Integer> list = new ArrayList<>();
    for (int i = 1; i <= n; i++) {
        if(n%i==0) list.add(i);
    }
    int[] answer = new int[list.size()];
    for(int i = 0; i < list.size(); i++) {
        answer[i] = list.get(i);
    }
    return answer;
}
}

```

Stream ?? ??

```

import java.util.*;
import java.util.stream.*;

class Solution {
    public int[] solution(int n) {
        return IntStream.rangeClosed(1, n) // 1~n 까지 모든 수
            .filter(i -> n % i == 0)      // n의 약수
            .toArray();                    // int[]로 변환
    }
}

```

List ? Array ???

List	Array
List<String>	list.toArray(new String[0])
List<Integer>	list.toArray(new Integer[0])
List<Integer> → int[]	list.stream().mapToInt().toArray()

? 1. List ? Array (???, ?: Integer, String ?)

- toArray() 返回一个 Object[] 类型的数组。

```
List<String> list = Arrays.asList("a", "b", "c");
String[] arr = list.toArray(new String[0]);
```

- list.toArray(new String[0]): 返回一个 Object[] 类型的数组。
- new String[0] 表示一个长度为 0 的字符串数组。

```
List<Integer> list = Arrays.asList(1, 2, 3);
Integer[] arr = list.toArray(new Integer[0]);
```

? 2. List ? Array (???, ?: int)

- Java 中 List<Integer> 和 int[] 都是不可变的。
- 因此，需要将 List 转换为 int[]。

```
List<Integer> list = Arrays.asList(1, 2, 3);
int[] arr = new int[list.size()];

for (int i = 0; i < list.size(); i++) {
    arr[i] = list.get(i); // 将 List 中的元素复制到数组中
}
```

Array ? List ???

	返回 List 的方法
<code>String[]</code>	<code>Arrays.asList(arr)</code>
<code>Integer[]</code>	<code>Arrays.asList(arr)</code>
<code>int[]</code>	<code>Arrays.stream(arr).boxed().collect(Collectors.toList())</code>

? 1. ??? ?? ? List (?: String[], Integer[] ?)

```
String[] arr = {"a", "b", "c"};
List<String> list = Arrays.asList(arr);
```

- `Arrays.asList()` 返回一个 `List` 视图，该视图直接引用数组的底层数据。
- 因此，通过该视图对 `List` 进行的修改会直接影响到原始的数组。

```
List<String> modifiableList = new ArrayList<>(Arrays.asList(arr));
```

? 2. 如何创建 List (?: `int[]`, `double[]` ?)

- 使用 `Arrays.asList()` 可以创建 `List<Integer>` 列表。
- `Stream + boxed()` 也可以创建 `List`。

```
int[] arr = {1, 2, 3, 4};

List<Integer> list = Arrays.stream(arr) // IntStream
                          .boxed()      // int → Integer (装箱)
                          .collect(Collectors.toList());
```

Revision #9

Created 19 May 2025 02:03:27 by Dain

Updated 23 May 2025 04:24:39 by Dain