First Program
The main method is called and its put on thhe stack.
Then it creates an “int[]” array named “numbers” with the values of (100, 200, 300, 400) in the heap, with a reference to it from the stack.
The printarray method is called with “numbers” as a reference. And a new stack for printArray is created on the stack. Then the numbers reference is used as a parameter.
Inside printArray, the “list” reference points to the same array as numbers, then prints “100 200 300 400 then \n, or a newline”
printarray completes, then it goes off the stack.
Then copyArray method is called, passing ‘numbers’ array again, which is put on the stack.
Inside copyArray, a new int[] array called newList is created with the same length as ‘list’ and its put int he heap, with a reference on the stack to copyArray.
The copyArray method goes thru ‘list’ array, copying the elements to ‘newList’
The copyArray method returns reference to newLIst, which is assigned to newNumbers, then copyArray goes off the stack.
doubleArrayContent method is called with numbers array ref to a new stack. Inside ‘doubleArraycontent’ the ‘list’ reference goes to the same array as ‘numbers’ then it doubles the numbers to get ‘200,400,600,800’. After this the method is done so off the stack.
Then printArray is called again with **numbers** as reference, and its put on the stack. Then the method prints ‘200 400 600 800’ then \n newline, and it is put off the stack.
The ‘printArray’ method is called for newNumbers array, and put on the stack. Then ‘newNumbers’ reference is used as a parameters, and then prints ‘100 200 300 400’.
Total is

```
100 200 300 400
200 400 600 800
100 200 300 400
```

Code 2
Main method is called, put on the stack
It creates an int[] array called list, with the elements (1,2,3,4,5) - this array is put on the heap with a reference to it in the main stack.
Then methodX is called with a list array reference. It’s put on the stack with list reference as a parameters. In methodX, a new int[] called newList is created with same length as list, then stored in the heap. The methodX goes thru list array and then copies in reverse order to get (5,4,3,2,1).
The methodX method is done so it goes off the stack.
Then printArray is called, using list array ref from the main method. It goes onto the stack with the list reference pushed as a parameter.
Inside printArray, the list ref points to eh original array and prints 1,2,3,4,5 with a newline

Output :

```
1 2 3 4 5
```