```
/* SELF ASSESSMENT

 1. Did I use easy-to-understand meaningful variable names formatted properly (in lowerCamelCase)?  My variable names were all named and formatted appropriately.

        Mark out of 5:  5

 2. Did I indent the code appropriately?  Yes, my code was indented correctly.

        Mark out of 5: 5

 3. Did I write the initialiseHighScores function correctly (parameters, return type and function body) and invoke it correctly?  I used the correct parameters as well as an appropriate return type.
         The function body was also neat as well as efficient.

       Mark out of 15:  15

 4. Did I write the printHighScores function correctly (parameters, return type and function body) and invoke it correctly?  I correctly used the parameters and function body. I felt it was unneeded to have a return.

       Mark out of 15:  15

 5. Did I write the higherThan function correctly (parameters, return type and function body) and invoke it correctly?  I correctly used the parameters, return type and function body. It was all invoked correctly.

       Mark out of 15:  15

 6. Did I write the insertScore function correctly (parameters, return type and function body) and invoke it correctly?  I used appropriate parameters, return type and a efficient function body.

       Mark out of 20:  20

 7. Did I write the main function body correctly (first asking for the number of scores to be maintained and then repeatedly asking for scores)?  My main body was wrote correctly. It first asks for the scores to keep and then asks for new scores infinitely.

       Mark out of 20:  20

 8. How well did I complete this self-assessment?  My self assessment was wrote to an appropriate level.

        Mark out of 5:  5

 Total Mark out of 100 (Add all the previous marks):  100

 */

import java.util.Scanner;

public class HighScores {

        public static void main(String[] args) {

                Scanner input = new Scanner ( System.in );
                boolean finished = false;

                System.out.println("Please insert the amount of "
                                + "High Scores you wish to store.");
                int numberOfHighScores = input.nextInt();
                double[] highScores = new double [ numberOfHighScores ];
                highScores = inistialisehighScores ( highScores );

                while ( !finished )
                {
                        printHighScores ( highScores);
                        System.out.println("Please enter a new number.");
                        double newNumber = input.nextDouble();
                        boolean higherThan = higherThan ( highScores, newNumber);
                        if ( higherThan)
                        {
                                highScores = insertScore ( highScores, newNumber);
                        }
                }
                input.close();

        }

        public static double[] inistialisehighScores ( double[] highScores)
        {
                for ( int counter = 0 ; highScores.length != counter ; counter++)
                {
                        highScores[counter] = 0;
                }
                return highScores;
        }

        public static void printHighScores ( double[] highScores)
```

```
	{
		System.out.print("The high scores are ");
		boolean finished = false;

		for ( int counter = 0; finished == false ; counter++)
		{
			if ( counter == highScores.length )
			{
				System.out.println();
				finished = true;
			}
			else if ( highScores[counter] == 0 )
			{
				System.out.println();
				finished = true;
			}
			else if ( counter == highScores.length - 1 )
			{
				System.out.print(highScores[ counter ]);
			}
			else
			{
				System.out.print(highScores[ counter ]+", ");
			}

		}
	}

	public static boolean higherThan ( double[] highScores, double newNumber)
	{
		if ( newNumber > highScores[highScores.length - 1])
		{
			return true;
		}
		else
		{
			return false;
		}
	}

	public static double[] insertScore ( double[] highScores, double newNumber)
	{
		double holdNumber = 0;

		for ( int counter = 0 ; counter < highScores.length ; counter++)
		{
			if ( newNumber > highScores[counter])
			{
				holdNumber = highScores[counter];
				highScores[counter] = newNumber;
				newNumber = holdNumber;
			}
		}
		return highScores;
	}
}
```