A simple Java program.

This is a simple Java program. Look at it carefully and understand the following points:

```
public class MyProgram{
        public static void main(String[] args) {
                System.out.print("Welcome to Sri Lanka");
        }
}
```

1. As a naming convention we start a class name with a capital letter, so if the class name contains two words we start the second word in a capital letter again. Spaces are not allowed.
   eg:- MyProgram

2. The starting point of the Java program is the main method. This is similar to the main () in C. The class contains the main method is called as the main class.
3. Now open a text editor and write the above Java program, to save the file:
   (I) The file name should be exactly same as the class name (case sensitive), here "MyProgram"
   (II) Use the file extension as .java, here MyProgram.java
4. To compile
   (I) Set the path
   (II) type javac MyProgram.java
   (III) Now you can see the class file (MyProgram.class) is created in the location that you have saved your MyProgram.java file,
5. To run
   (I) type java MyProgram

Other technical points:

1. Java application can have any number of classes, but only one of them MUST include a main method,
2. public: this is a keyword and an access modifier. This keyword declares the main method as unprotected and therefore accessible to all the other classes (otherwise it is not possible to start the program)
3. static: this declares a method as one that belongs to the class, not a part of an object. The main method must always be declared as static because the compiler uses this method as the starting point, even before creating any objects.
4. void: declare that the main method does not return any value.

Some questions:

1. what are the keywords in the above program?
2. what are the identifiers in the above program?
3. what are the separators in the above program?
4. what is a literal in the above program?

Structure of a generic Java program.

1. Documentation section

- contains set of comment lines
- describing name of the program, purpose, author, version and other useful information
- to start a comment \\ is used, or comments are written between * and *\

eg:- \\My first Java program
My first Java program\

2. Package statements
- informs the compiler that the class defined here belongs to a particular package
- the first statement allowed in Java
- for example lets take a package called myPackage

eg:-

```
package myPackage;
public class MyProgram{
        public static void main(String[] args) {
                System.out.print("Welcome to Sri Lanka");
        }
}
```

compile this and save the MyProgram.class file in a folder called myPackage. Later on this this package can be imported into another java program and this class can be reused.

to import this class to another program:
```
import myPackage.MyProgram;
```

to import all the classes of myPackage:
```
import myPackage.*;
```

3. Import statements

- import the relevant libraries, this is similar to #include in C
- between package statement and class definitions

eg:-

```
package myPackage2;

import Java.lang.*; //importing a built in package
import Java.io.*; //importing a built in package
import myPackage.*; //importing a user defined package

public class MyProgram{
        public static void main(String[] args) {
                System.out.print("Welcome to Sri Lanka");
        }
}
```

4. Interface statements/class definitions

Interface

- more abstract description of a class
- if a class contains at least one unimplemented method it is an interface

eg:-

```
public interface Student{
        void display (); //unimplemented method
}
```

Class
eg:-

```
public class MyProgram{
        public void test() {
                System.out.print("Welcome to Sri Lanka");
        }
}
```

Main Class

- this is the class containing the main method
- main method is the starting point of the program