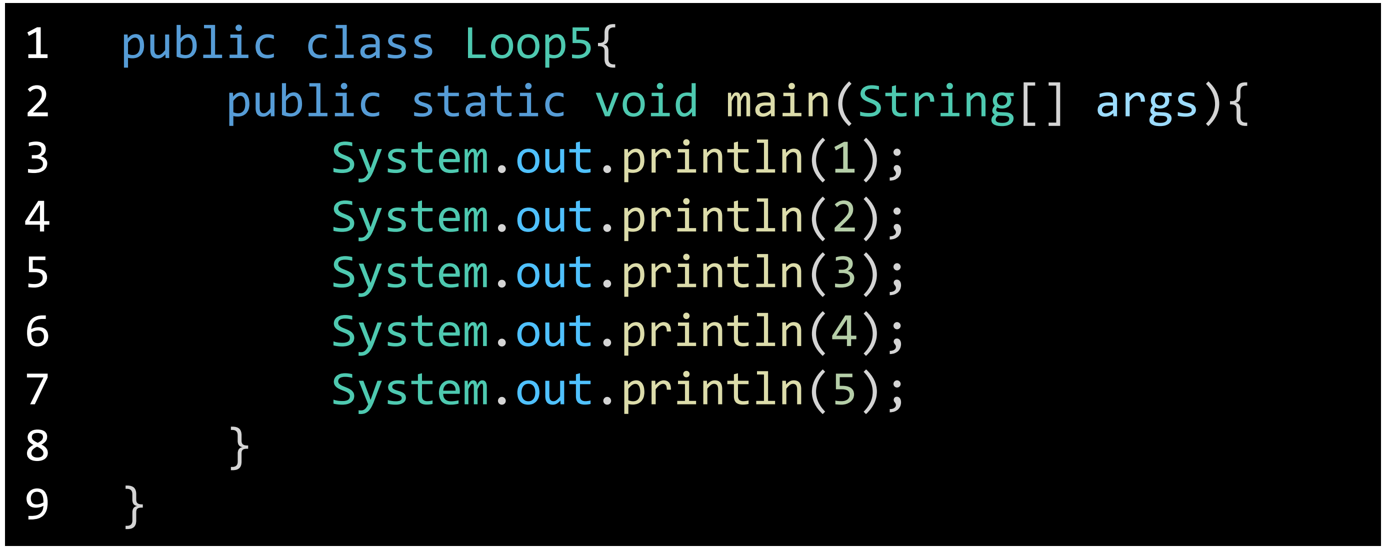
```
mirror_mod = modifier_ob.
set mirror object to mirror
mirror_mod.mirror_object
operation == "MIRROR_X":
mirror_mod.use_x = True
mirror_mod.use_y = False
mirror_mod.use_z = False
 _operation == "MIRROR_Y":
mirror_mod.use_x = False
mirror_mod.use_y = True
mirror_mod.use_z = False
 _operation == "MIRROR_Z":
mirror_mod.use_x = False
mirror_mod.use_y = False
```

Topic 8

Loops

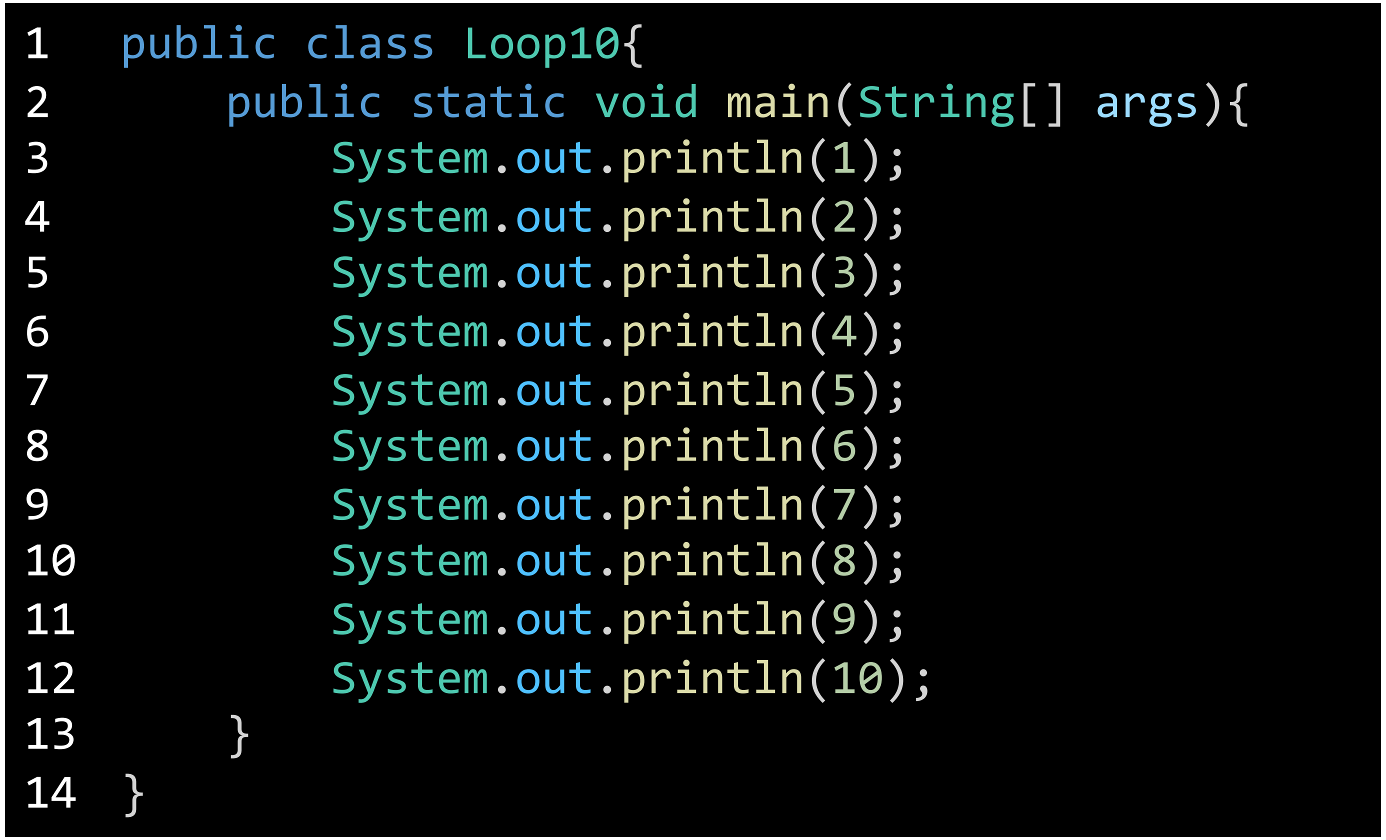
Loops

Design a program that prints the numbers from 1 to 5

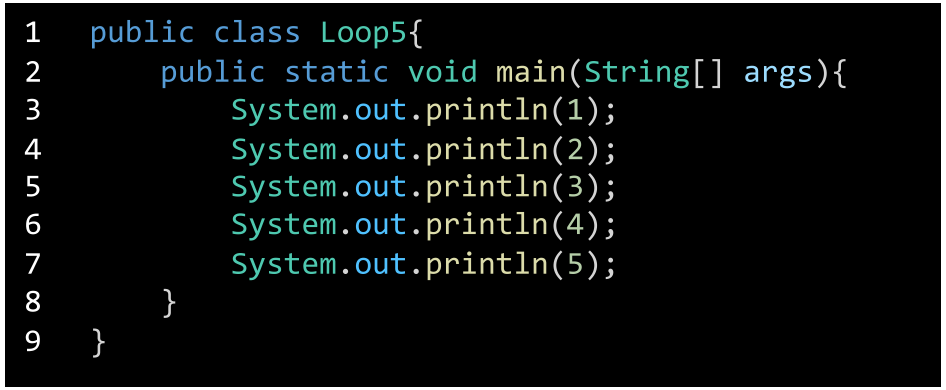


```java
public class Loop5{
    public static void main(String[] args){
        System.out.println(1);
        System.out.println(2);
        System.out.println(3);
        System.out.println(4);
        System.out.println(5);
    }
}
```

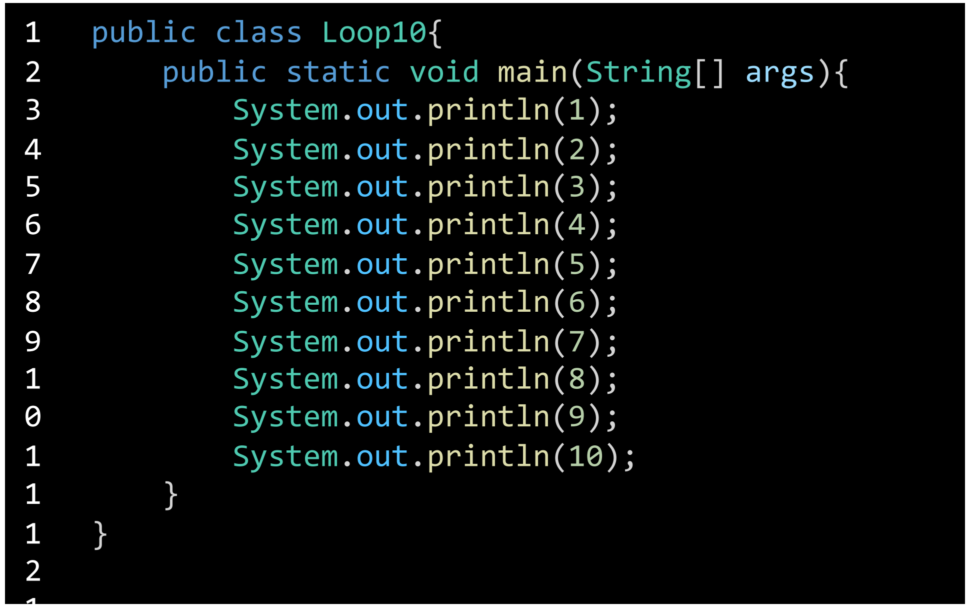
Design a program that prints the numbers from 1 to 10



```java
public class Loop10{
    public static void main(String[] args){
        System.out.println(1);
        System.out.println(2);
        System.out.println(3);
        System.out.println(4);
        System.out.println(5);
        System.out.println(6);
        System.out.println(7);
        System.out.println(8);
        System.out.println(9);
        System.out.println(10);
    }
}
```

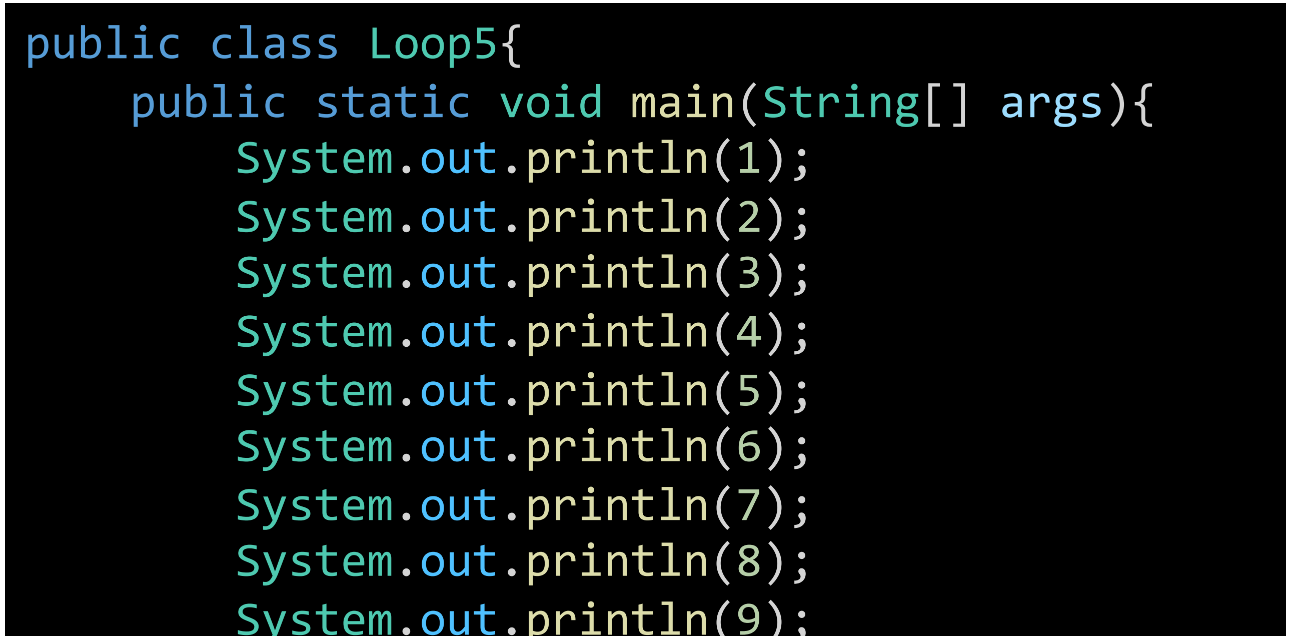


```java
public class Loop5{
    public static void main(String[] args){
        System.out.println(1);
        System.out.println(2);
        System.out.println(3);
        System.out.println(4);
        System.out.println(5);
    }
}
```



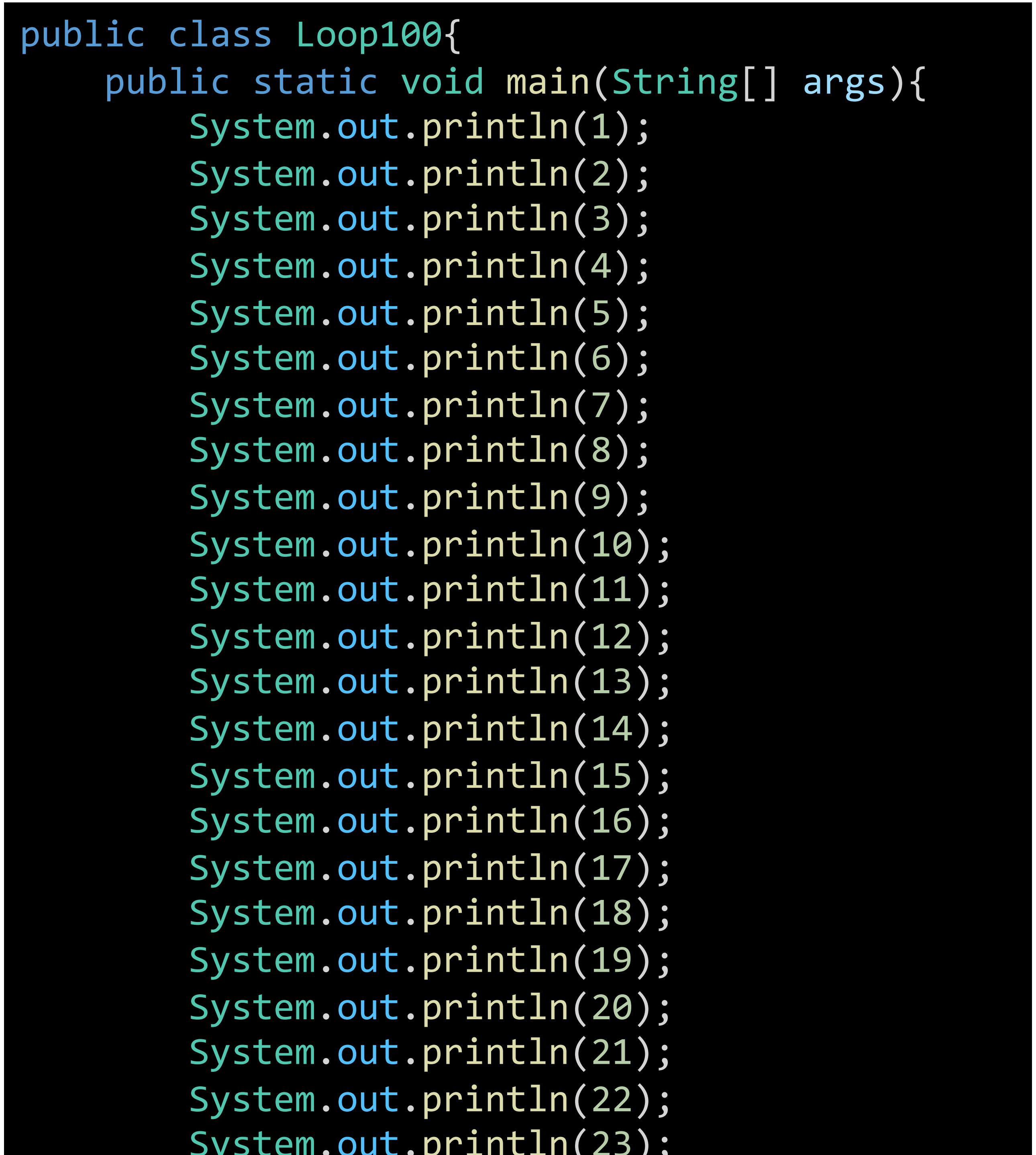
```java
public class Loop10{
    public static void main(String[] args){
        System.out.println(1);
        System.out.println(2);
        System.out.println(3);
        System.out.println(4);
        System.out.println(5);
        System.out.println(6);
        System.out.println(7);
        System.out.println(8);
        System.out.println(9);
        System.out.println(10);
    }
}
```

Now one that prints the numbers from 1 to 100



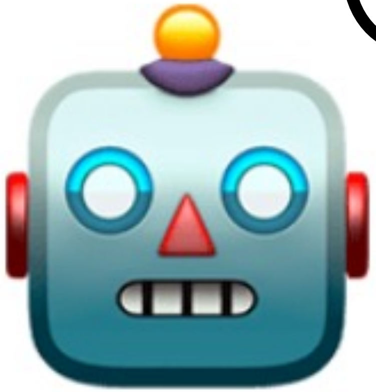
```java
public class Loop5{
    public static void main(String[] args){
        System.out.println(1);
        System.out.println(2);
        System.out.println(3);
        System.out.println(4);
        System.out.println(5);
        System.out.println(6);
        System.out.println(7);
        System.out.println(8);
        System.out.println(9);
```

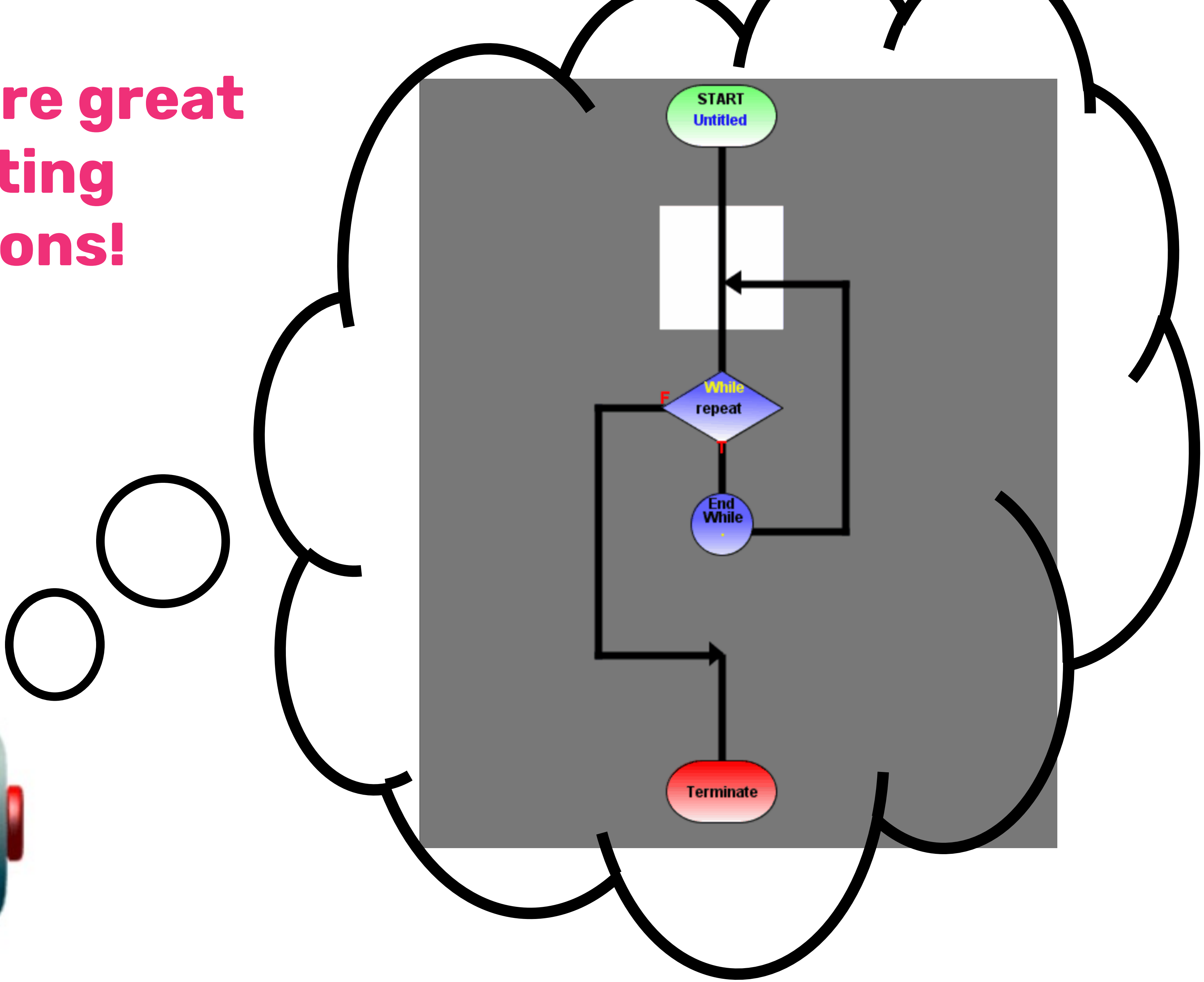
Now one that prints the numbers from 1 to 100



```
public class Loop100{
    public static void main(String[] args){
        System.out.println(1);
        System.out.println(2);
        System.out.println(3);
        System.out.println(4);
        System.out.println(5);
        System.out.println(6);
        System.out.println(7);
        System.out.println(8);
        System.out.println(9);
        System.out.println(10);
        System.out.println(11);
        System.out.println(12);
        System.out.println(13);
        System.out.println(14);
        System.out.println(15);
        System.out.println(16);
        System.out.println(17);
        System.out.println(18);
        System.out.println(19);
        System.out.println(20);
        System.out.println(21);
        System.out.println(22);
        System.out.println(23);
```

Computers are great at repeating instructions!





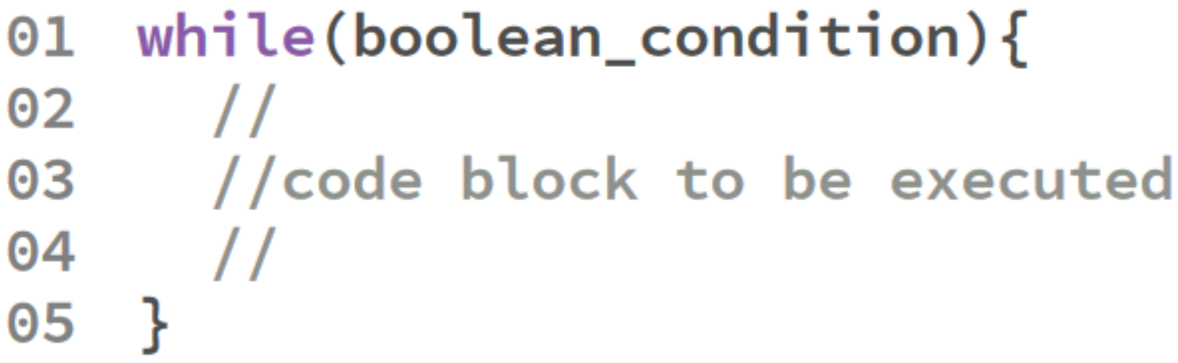
Loops

A block of instructions that will repeat based on a boolean expression.

1. **Loop body**: Code to be repeated
2. **Control variable**: Variable that will control
3. **Exit condition**: When will the block of code stop repeating?

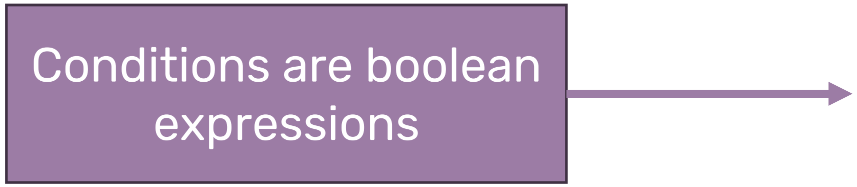
While loops

Sintaxis

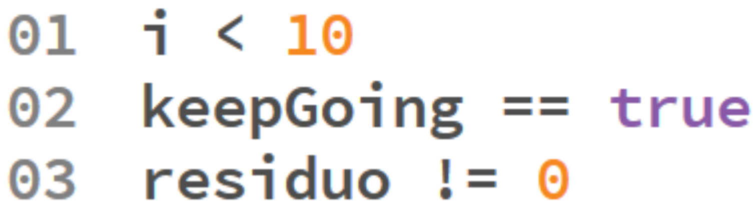


```
while(boolean_condition){
  //
  //code block to be executed
  //
}
```

Examples of boolean conditions



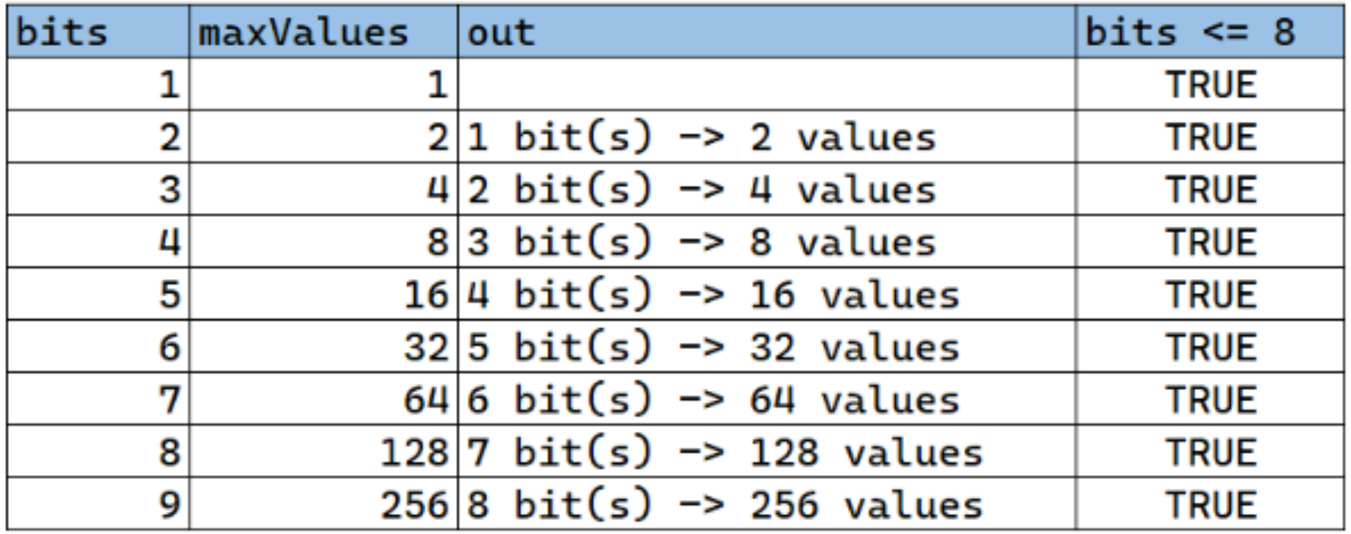
Conditions are boolean expressions



```
i < 10
keepGoing == true
residuo != 0
```

```java
public class LoopTest {

  public static void main(String[] args) {

    int bits = 1;
    int maxValues = 1;

    while(bits <= 8) {
      maxValues *= 2;
      System.out.println(bits + " bit(s) -> " + maxValues + " values.");
      bits++;
    }
  }

}
```

Code trace



bits	maxValues	out	bits <= 8
1	1		TRUE
2	2	1 bit(s) -> 2 values	TRUE
3	4	2 bit(s) -> 4 values	TRUE
4	8	3 bit(s) -> 8 values	TRUE
5	16	4 bit(s) -> 16 values	TRUE
6	32	5 bit(s) -> 32 values	TRUE
7	64	6 bit(s) -> 64 values	TRUE
8	128	7 bit(s) -> 128 values	TRUE
9	256	8 bit(s) -> 256 values	TRUE

initial values -->

<-- Enters while loop

<-- Exit condition is met

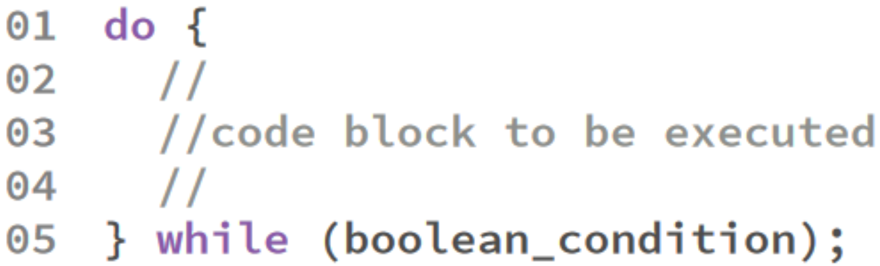
Do-while

```
do {
  //
  //code block to be executed
  //
} while (boolean_condition);
```

Do-While

Do-whiles are very similar to whiles, with the exception that it will always be executed at least once.

This is because the Boolean condition is evaluated at the end of the loop, instead of at the start. This loops are very useful when validating inputs from the keyboard.



```
do {
  //
  //code block to be executed
  //
} while (boolean_condition);
```

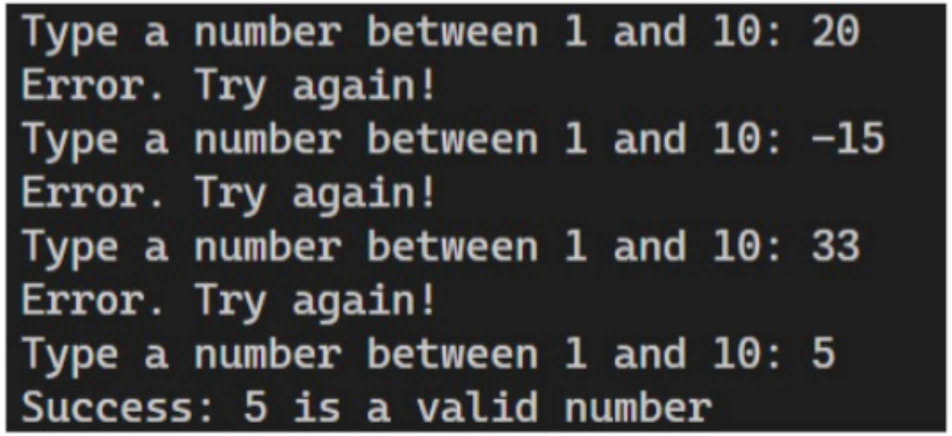
```java
boolean invalidInput = true;
int i;
Scanner keyboard = new Scanner(System.in);
do {
    System.out.print("Type a number between 1 and 10: ");
    i = keyboard.nextInt();
    if (i >= 1 && i <= 10){
        invalidInput = false;
    } else {
        System.out.println("Error. Try again!");
    }
} while(invalidInput == true);

System.out.println("Success: " + i + " is a valid number");

keyboard.close();
```

```java
boolean invalidInput = true;
int i;
Scanner keyboard = new Scanner(System.in);
do {
    System.out.print("Type a number between 1 and 10: ");
    i = keyboard.nextInt();
    if (i >= 1 && i <= 10){
        invalidInput = false;
    } else {
        System.out.println("Error. Try again!");
    }
} while(invalidInput == true);

System.out.println("Success: " + i + " is a valid number");

keyboard.close();
```

```java
boolean invalidInput = true;
int i;
Scanner keyboard = new Scanner(System.in);
do {
    System.out.print("Type a number between 1 and 10: ");
    i = keyboard.nextInt();
    if (i >= 1 && i <= 10){
        invalidInput = false;
    } else {
        System.out.println("Error. Try again!");
    }
} while(invalidInput == true);

System.out.println("Success: " + i + " is a valid number");

keyboard.close();
```

```java
boolean invalidInput = true;
int i;
Scanner keyboard = new Scanner(System.in);
do {
    System.out.print("Type a number between 1 and 10: ");
    i = keyboard.nextInt();
    if (i >= 1 && i <= 10){
        invalidInput = false;
    } else {
        System.out.println("Error. Try again!");
    }
} while(invalidInput == true);

System.out.println("Success: " + i + " is a valid number");

keyboard.close();
```

```java
boolean invalidInput = true;
int i;
Scanner keyboard = new Scanner(System.in);
do {
    System.out.print("Type a number between 1 and 10: ");
    i = keyboard.nextInt();
    if (i >= 1 && i <= 10){
        invalidInput = false;
    } else {
        System.out.println("Error. Try again!");
    }
} while(invalidInput == true);

System.out.println("Success: " + i + " is a valid number");

keyboard.close();
```



```
Type a number between 1 and 10: 20
Error. Try again!
Type a number between 1 and 10: -15
Error. Try again!
Type a number between 1 and 10: 33
Error. Try again!
Type a number between 1 and 10: 5
Success: 5 is a valid number
```

For loop

For loops

For-loops are generally used to repeat a set of instructions a set number of times.

The syntax is as follows:

```
for(initialization; condition; update) {
   //body of loop to be repeated
}
```

```java
for (int i = 9; i >= 0; i = i - 2) {
    System.out.println(i);
}
```

Example

Parrot Salute

In the Monterrey zoo, there is a very well-educated parrot that salutes everyone that passes by.

Build a program that reads the group size from the keyboard, and then print a salute for every persons in the group.

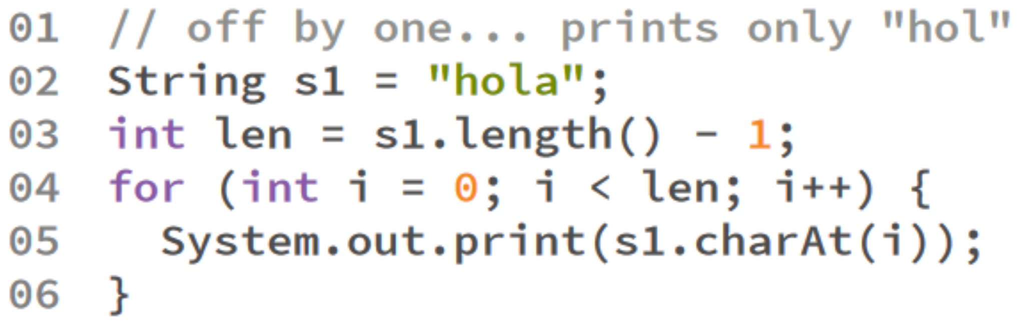
Consider that groups need to be between 0 and 10 in size.

```
Scanner keyboard = new Scanner(System.in);

System.out.print("How big is your group?: ");
int groupSize = keyboard.nextInt();

while (groupSize <= 0 || groupSize >= 11){
    System.out.print("Hmm that doesn't seem right. How big is your group: ");
    groupSize = keyboard.nextInt();
}

String salute = "";
for(int i = 0; i < groupSize; i++){
    salute = salute + "Hi";
    if (i < (groupSize-1)){
        salute = salute + "-";
    }
}

System.out.println(salute);
```

Common problems

Common loop problems

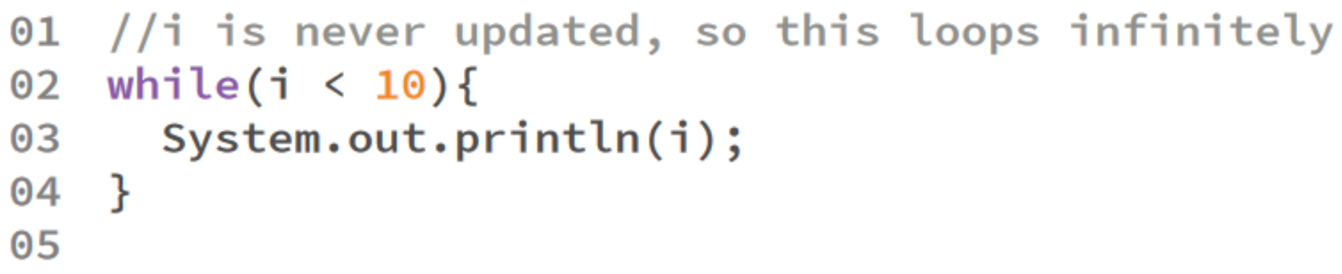
1. **Off by one:** The program loops fewer times than expected, or more times than expected.



```java
// off by one... prints only "hol"
String s1 = "hola";
int len = s1.length() - 1;
for (int i = 0; i < len; i++) {
  System.out.print(s1.charAt(i));
}
```

Common loop problems

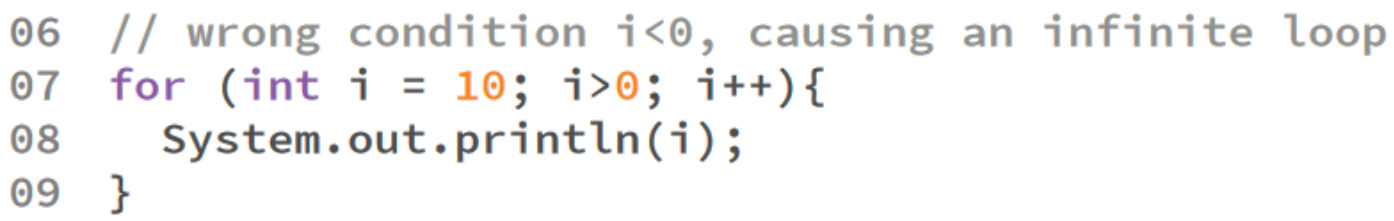
2. Infinite loops: Under certain conditions, the loop will never end



```
//i is never updated, so this loops infinitely
while(i < 10){
  System.out.println(i);
}

```

Common loop problems

3. Wrong boolean expressions: The loop is never entered or never exited



```java
// wrong condition i<0, causing an infinite loop
for (int i = 10; i>0; i++){
  System.out.println(i);
}
```