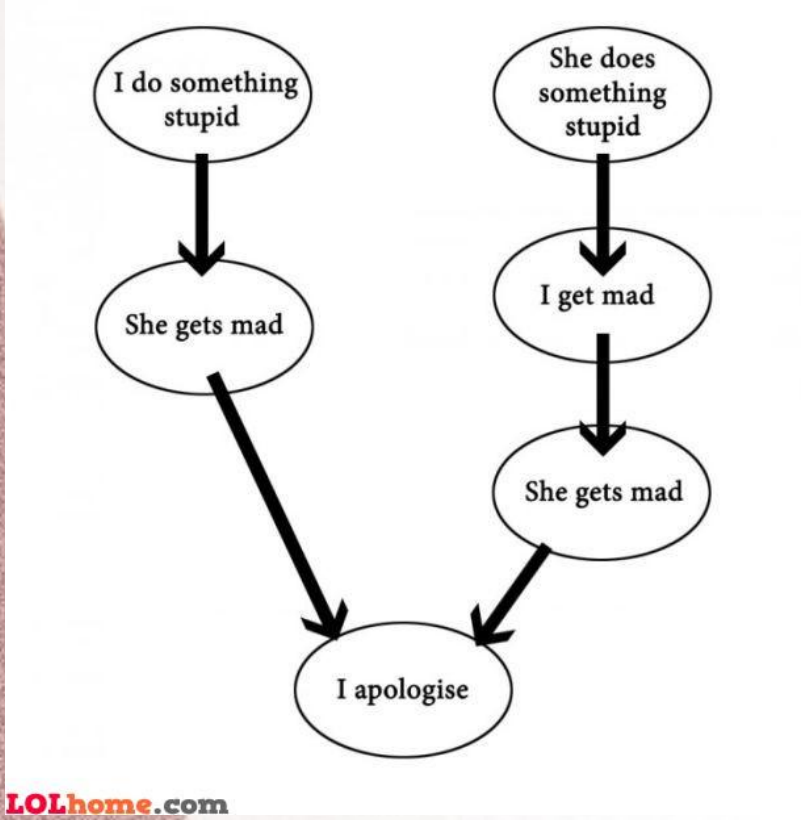
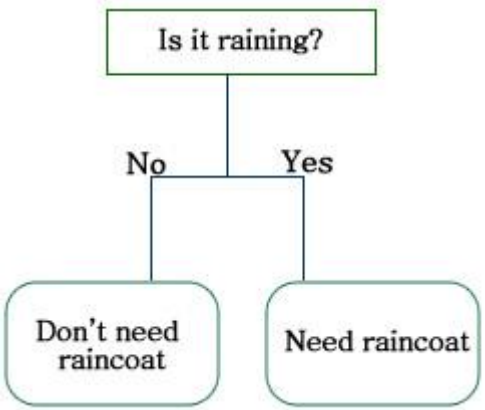
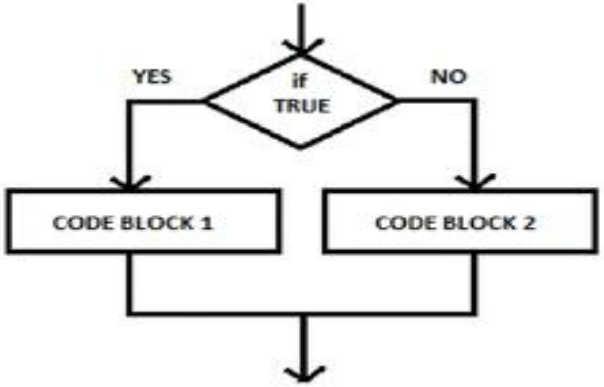
Chapter 3 – Flow of control



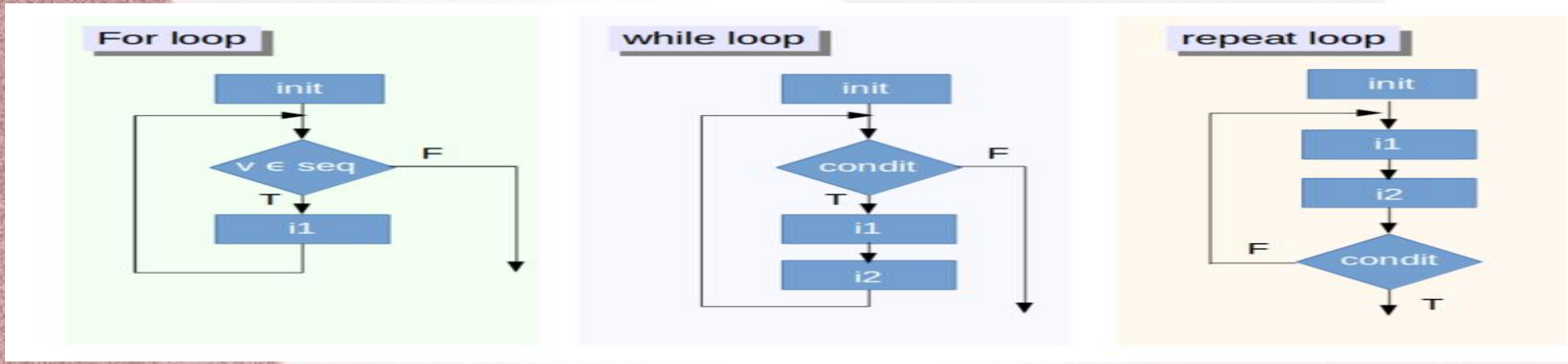
Flow of control

- Branching mechanism(is like a tree): if-else, if , switch





- Loops(basically re-doing statement again): while, do-while, for



Operator used for comparing

Symbol	Name
==	equal
!=	Not equal
>	Bigger than
<	Less than
>=	Bigger than or equal
<=	Less than or equal

Let A = 10
B = 20

Expression	Result
A == B	False
A != B	True
A > B	False
A < B	True
A >= B	False
A <= B	False

If statement

Check if condition(must be boolean || “compaison”) is true, if it is true then it will do the statement else it move on(Skip)

```
If ( Condition ){
Statement;

}
```

If statement example

```
int randomInt = 25;

if(randomInt > 25)
   System.out.println("twinkle");
   System.out.println("twinky");
System.out.println("twoeley");
```

Compound statement

- If we have more than one statement for the “if statement”, we need to enclose it with a bracket { }
- *if we do not enclose it, it will only do the first statement and ignore the rest

Example:

```
int x = 10;
if(x>20)
System.out.println(“x is bigger than 20”);
System.out.println(“This is the end :O”);
```

Only Statement will execute

Compound statement example

Bad compound

```
int weight = 10;
if(weight > 200)
        System.out.println("You are so fat!");
        System.out.println("Good bye");
```

This example show that it will skip the first if statement since 10 is not bigger than 200

It will only print “Good bye”
Note: use { } to fix the bad compound

If-else statement

- Similar to if statement but if-else allow more flexibility because there are two alternate ways
- Example:

```
If (weight < 200)
System.out.println("eat more boy");
Else
System.out.println("eat less");
```

More on if-else

- If-else always match to closest if and else
- Example:

```
If(condition 1)
if(condition 2)
else
```

The else will go with the second if statement rather than the first one

If-else statement example

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

System.out.println("Please enter your age");

int age = kb.nextInt();
if(age > 18){
System.out.println("you are legal!");
}
else
System.out.println("you are not legal yet!");
```

What if we enter age as 18?

What if we enter age as ???

Switch statement - Syntax

```
Switch ( Expression) {
Case value1:
Statement1;
Break;

Case value2:
Statement2;
Break;
:
:
Case valueN;
StatementN;
Break;

Default:
Default-statement
Break;
}
```

It is the same as an if-else statement but made more compact and easier to see.

You can always convert switch to if-else and vice versa

Note:
*it is hard to convert an switch <-> if-else with greater or less than

*after Case, if value is numeric there is no quotation, else we use ' ' for character

Switch example

They are equal

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

int input = kb.nextInt();

switch(input){

case 1:
System.out.println("user entered 1");
break;

case 2:
System.out.println("user entered 2");
break;

default:
System.out.println("user entered
something other than 1 & 2");
break;
```

```
Scanner kb = new Scanner(System.in)

int input = kb.nextInt();

if(input == 1){
System.out.println("user entered 1"
}
else if(input == 2){
System.out.println("user entered 2"
}
else
System.out.println("user entered
 something other than 1 & 2");
```

Rewrite switch and if-else

Switch into if-else

```
int age = 20;

switch(age){

case 18:
System.out.println("you are 18 year
old");
break;

case 19:
System.out.println("you are 19 year
old");
break;

default:
System.out.println("we don't know
how old you are!");
break;
}
```

If-else into switch

```
Scanner kb = new
Scanner(System.in);
int weight = kb.nextInt();

if(weight == 200)
System.out.println("you are
fat, go exercise");
else if(weight == 100)
System.out.println("you are
skinny, eat more");
else
System.out.println("I don't
know anymore");
```

Rewrite switch and if-else || answers

if-else

```
int age = 20;

if(age == 18)
System.out.println("you are 18 year old");
else if(age == 19)
System.out.println("you are 19 year old");
else
System.out.println("we don't know how old you are!");
```

Switch

```
Scanner kb = new Scanner(System.in);
int weight = kb.nextInt();

switch(weight){
case 200:
System.out.println("you are fat, go exercise");
break;

case 100:
System.out.println("you are skinny, eat more");
break;

default:
System.out.println("I don't know anymore");
break;
```

Switch and break - Guess the output

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

String userInput = kb.next();
char aChar = userInput.charAt(0);

switch(aChar){

case 'x':
System.out.print("x");
break;

case 'y':
System.out.print("y");

case 'z':
System.out.print("z");


default:
System.out.print("0");
break;

}
```

What if we entered 'x'

What if we entered 'z'

Switch and break - Guess the output

```
int price = 6;
switch (price) {
case 2: System.out.println("It is: 2");
default: System.out.println("It is:
default");
case 5: System.out.println("It is: 5");
case 9: System.out.println("It is: 9");

}
```

Conditional operator

It's basically a shortcut to the if-else statement

Syntax:

Condition ? Expression1 : Expression 2

-if condition is true then execute expression1 else we execute expression 2

Example:

```
Int num1 = 20, num2 = 10;
Int num = ((( num1 >num2) ? num1:num2 );
System.out.println(num);
```

While loop

Syntax while(condition){

statement

}

Algorithm of Success

```
while(noSuccess)
{
        tryAgain();

        if(Dead)
                break;
}
```

- While the condition is true, it will keep re-looping all the statement till it is false
- Watch out for infinite loop, you should have a condition that reach to end

While loop - Example

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

int userInput = kb.nextInt();

while(userInput>0)
{
System.out.println(userInput);
userInput--;
}
```

1) What if user entered 8?

More about while loop

1)
```
int remainingAt = 5;

while(remainingAt>0){
System.out.println("@");
remainingAt--;
}
```

2)
```
int remainingAt = 5;

while(remainingAt>0)
System.out.println("@");
remainingAt--;
```

Do-while loop

Syntax:

```
do{
statement
}
while(condition);
```

- Same concept as while loop but will execute statement at least once

Do-while vs while example – guess the output

Do-While loop

```
int i = 0;
do{
System.out.println("Executed "+ i + " time");
i--;
}
while(i > 0);
```

While loop

```
int n = 0;
while(n>0){
System.out.println("Executed " + n);
n--;
}
```

For loop

-for loop is useful if you know how many iteration you want to loop

Syntax:

Initialize variable

Update the variable

```
for(int iteration; iteration <numTime; iteration++){
Statement
}
```

Do loop if condition is true

Example:

```
for(int i =0; i<5; i++){
System.out.println("hello");
}
```

Will print "hello" 5 times

For loop - example

1)

```
int y = 0;
for (int i = 0; i<10; ++i) {
  y += i;
}
System.out.println(y);
```

What's the output of the for loop?

2)

```
int y = 0;
for (int i = 0; i<10; ++i) {
  y += i;
}
```

What's “i” after the for loop?

Nested

A nested is like a package inside another package,

Example of nested if statement

```
int apple = 10;
int bacon = 15;
if(apple > 0){
System.out.println("I eat apple");
if(bacon < 10){
System.out.println("I also love bacon");
}

}
else
System.out.println("I don't eat
anything");
```

We can also have a nested If-else, for, while, do-while loops

Logical Operator

And (&&)

-Only true if both is true and false if one is false

Example:

A	B	A && B
true	true	
true	false	
false	true	
false	false	

Logical Operator

Or (||)

-always true if at least one is true, false if both condition are false

Example:

A	B	A || B
true	true	
true	false	
false	true	
false	false	

Logical Operator

Not (!)

This will negate into opposite sign

Example:

!(true) -> will become false

!(false) -> will become true

A	!A
!True	False
!False	True

Logical Operator - Example

```
int weight;
boolean valid = false;

Scanner kb = new Scanner(System.in);

do{
System.out.println("what's your weight?");
weight = kb.nextInt();

if( (weight> 0) && (weight< 1000) ){
System.out.println("you weight: " + weight);
}else
System.out.println("Invalid please try again!\n");

}while(!valid);
```

Break

- Break: break will end the enclosing loop

Example:

```
for(int i = 0; i < 10; i++){
    if(i==4){
    break;
    }
    System.out.println(i);
}
```

Continue

- Continue: Will end the current loop and continue the loop.

Example:

```
for(int i = 1; i <= 100 ; i++){
    if(i % 2 == 0)
        continue;
    System.out.println(i);
   }
```

Exit

- When we want to end a program, we can use “exit(0);” to close the program.
- The 0 means the program terminate normally

Note: used to find what type of errors

```
boolean quit = false;

if(quit){
System.exit(0);
}
```