Programming

Intermediate I

1

Variables

variables

A part of the memory that has a value.

variables

typeVariable **nameVariable**

example: int **age**

variables

we can add value

```
int age

age = 15
```

variables

we can start with a value

```
int age = 15
```

variables

we can start with a value

```
int two= 1 + 1

int seven= 10 - 3

int four= 2 * 2

int ten=  40 / 4
```

variables

```
int one= 5 % 2
```

5 / 2 = 2,5

But we expect an integer variable

i.e.

5/2 = 2 with rest 1

the symbol of % means remainder of a division

variables

calculate the age next year

```
int ageNextYear, age;
age = 10
ageNextYear= edagead + 1;
```

variables

a small program

```
int ageNextYear, age

age= 20


ageNextYear= age+ 1


print ageNextYear
```

PRINT = print in a screen

a small Java program

variables

```
class SumAges{

    public static void main(String[] args) {

        int age= 20;

        System.out.println(age);

        int ageNextYear;

        ageNextYear= age+ 1;

        System.out.println(ageNextYear);

    }

}
```

variables

small Java program

```
class SumAges {
  public static void main(String[] args) {
    int age = 20;
    System.out.println(age);
    int ageNextYear;
    ageNextYear= age+ 1;
    System.out.println(ageNextYear);
  }
}
```

Command needed to start a program in JAVA

PRINT = print in a screen

JAVA needs a semicolon

variables

```
class SumAges{
    public static void main(String[] args) {
        int age= 20;
        System.out.println(age);
        int ageNextYear;
        ageNextYear= age+ 1;
        System.out.println(AgeNextYear);
    }
}
```

variables

a small Python program

```
if __name__ == '__main__':

        age= 20

        print(age)

         ageNextYear= age+ 1

        print(ageNextYear)
```

variables

```
if __name__ == '__main__':

    age= 20

    print(age)

    ageNextYear= age + 1

    print(ageNextYear)
```

Command needed to start a program in PYTHON

variables

Other types of variables

```
double pi = 3.14

boolean underAge = age<=18

char letter = 'a'

string name= 'John Connor'
```

2

Operators- part 1 - < > <= >=

OPERATORS

> Greater than

< Less than

> = Greater or equal

< = Less or equal

= equal

!= different

OPERATORS

```
int age= 22

if age > 18

        print "Legal age to drive"

end_if
```

Can drive

```
int age= 15

if age < 18

        print "Underage to drive"

end_if
```

Can NOT drive

OPERATORS

```
int age= 22

if age >= 18

        print "Legal age to drive"

end_if
```

Can drive

```
int age= 15

if age < = 17

        print "Underage to drive"

end_if
```

Can NOT drive

OPERATORS

```
int age= 22

if age = 18

        print "Legal age to drive"

end_if
```

Can drive

```
int age= 15

if age != 18

        print "Underage to drive"

end_if
```

Can NOT drive

OPERATORS

```
int  age = 19

int fines= 10

if age >=  18

        print "Can Drive"

end_if
```

Here we have a problem, we need to validate more things.
In other words, it is not enough to be over 18 years old, you cannot have fines as well.

3

Operators - part 2 - && || = !=

OPERATORS

&& and

|| or

= equal

&& - (and)

Think of a situation like:

I have to be over 18 years old and never have to be fined to have a driving license.

If I don't have any of them, I can't get my driving license

OPERATORS

```
int age= 19

int fines =  0

if age>= 18 && fines < 1

        print "Can Drive"

end_if
```

→ Can Drive

```
int age= 25

int fines =  8

if age>= 18 && fines < 1

        print "Can Drive"

end_if
```

→ Can NOT Drive

&&

If one part of the equation is false, we don't even need to read the other side.

The AND operator, to work, needs both parts to be TRUE.

OPERATORS

```
int age= 18
int fines =  9
if age>= 18 && fines < 1
        print "Can Drive"
end_if
```

```
int age= 23
int fines =  0
if age>= 18 && fines < 1
        print "Can Drive"
end_if
```

```
if age>= 18  &&  fines < 1
        print "Can Drive"
end_if
```

```
if age>= 18  &&  fines < 1
        print "Can Drive"
end_if
```

|| - (or)

Think of a situation like:

Today I go to work by bus or car.

If I don't have any of the parts it doesn't matter. Just 1 be true that I can get to my job

OPERATORS

```
int age= 30

int fines = 4

if age < 18 || fines >0

        print "Can NOT Drive"

end_if
```

→ Can NOT Drive

```
int age= 12

int fines = 0

if age < 18 || fines >0

        print " Can NOT Drive"

end_if
```

→ Can NOT drive

||

If one part of the equation is false, it doesn't matter.
The OR operator needs only one part to be TRUE to work.

OPERADORES

```
int age= 23

int fines =  9

if age<= 18 || fines < 1

        print " Can NOT  Drive"

end_if
```

```
int age= 43

int fines =  9

if age<= 18 || fines < 1

        print "Can NOT Drive"

end_if
```

```
if age<= 18   ||  fines < 1

        print "Can NOT  Drive"

end_if
```

```
if age<= 18   ||  fines < 1

        print "Can NOT Drive"

end_if
```

OPERATORS

```
int age= 19

int fines = 10

if age >=  18

        print "Can Drive"

end_if
```

Here we have a problem.
Missing a part.
What should we do if the age is less than 18?

4

IF and ELSE

IF AND ELSE

```
int age= 18
if age> = 18
        print "Can Drive"
end_if
```

```
int age= 18
if age> = 18
        print "Can Drive"
else
        print " Can NOT Drive"
end_if
```

→ Can Drive

ELSE
if the IF line is false, the program ignores it and looks for the other option, which is ELSE

IF AND ELSE

```
int age= 18
if age> = 18
        print "Can Drive"
else
        print " Can NOT Drive"
```

→ Can Drive

```
int age= 14
if age> = 18
        print "Can Drive"
else
        print " Can NOT  Drive"
```

→ Can NOT Drive

5

WHILE

WHILE

```
int age = 15
while (age< 18)
        print (age)
        age= age+ 1
```

```
15
16
17
```

```
int i = 0
while (i < 10)
    print(i)
    i = i + 1
```

```
0
1
2
3
4
5
6
7
8
9
```

WHILE
is a loop that repeats a piece of code as long as the condition is true.

WHILE

```
int age = 15
while (age  < 18)
        print (age )
        age = age + 1
```

```
int age = 15
while (15 < 18)
        print (15)
        age = age + 1
```

16

```
while (16 < 18)
        print (16)
        age = age + 1
```

17

```
while (17 < 18)
        print (17)
        age = age + 1
```

18

```
while (18 < 18) ✗
        print (age )
        age = age + 1
```

6

FOR

FOR

```
for (int a = 0; a< 4; a = a + 1)

    print ("Hello")
```

```
Hello
Hello
Hello
```

```
int a = 0
while (a < 4)

        print ("Hello")

        a = a + a
```

FOR

The idea is the same as while: make a piece of code be repeated while a condition remains true.

In addition you can initialize variables and the modifier of them.

FOR

```
for (int a = 0; a< 4; a = a + 1)
    print ("Hello")
```

```
for (int a = 0; a< 4; a = a + 1)
    print ("Hello")
```

a = 1

Hello

```
for (int a = 0; a< 4; a = a + 1)
    print ("Hello")
```

a = 2

Hello

```
for (int a = 0; a< 4; a = a + 1)
    print ("Hello")
```

a = 3

Hello

```
for (int a = 0; a< 4; a = a + 1)
    print ("Hello")
```

a = 4

7

CODE BLOCKS

Code Blocks

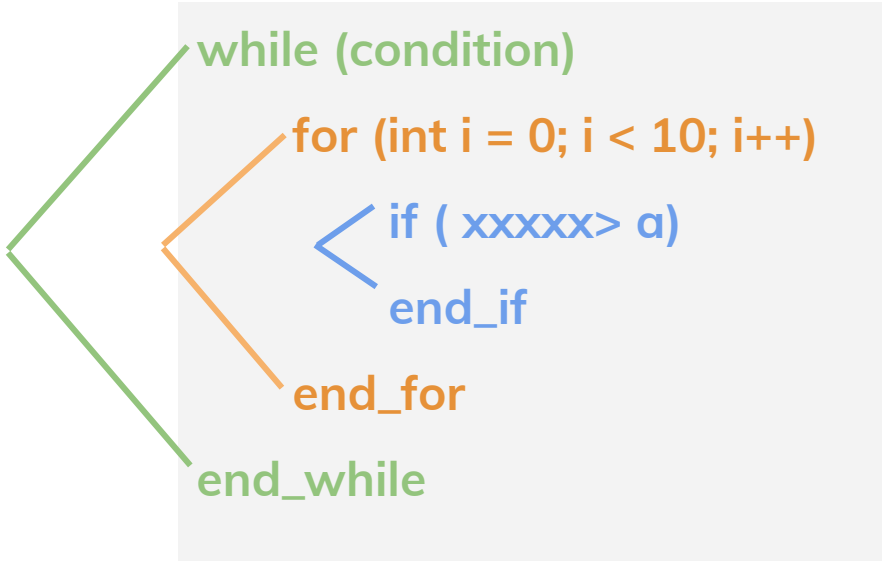
```
while (a < 4)

        print ("Hello")

        a = a + a

end_while
```

```
if age > = 18 {

        print "Can drive"

else

        print "Can NOT drive"

}
```

In some programming languages we use braces to open and close code blocks

Code Blocks

```
while (condition)
    for (int i = 0; i < 10; i++)
        if ( xxxxx> a)
        end_if
    end_for
end_while
```



We can have blocks of code within other blocks.

THANKS