A+ Computer Science

MATH / OOP

Calculations

Expressions

average = total / 5

sum = one + two

Expressions usually consist of operators, variables, and/or values.

Operators

+	addition
-	subtraction
*	multiplication
/	division
%	modulus

Integer Math

```
out.println("6 + 5 == " + (6+5));
out.println("6 - 5 == " + (6-5));
out.println("6 * 5 == " + (6*5));
out.println("6 / 5 == " + (6/5));
```

OUTPUT

```
6 + 5 == 11
6 - 5 == 1
6 * 5 == 30
6 / 5 == 1
```

Real Math

```
out.println("6.1 + 5.2 = = " + (6.1+5.2));
out.println("6.1 - 5.2 = = " + (6.1-5.2));
out.println("6.1 * 5.2 = = " + (6.1*5.2));
out.println("6.1 / 5.2 = = " + (6.1/5.2));
```

OUTPUT

```
6.1 + 5.2 = = 11.3
6.1 - 5.2 = = 0.8999
6.1 * 5.2 = = 31.72
6.1 / 5.2 = = 1.17307
```

intmath.java
realmath.java

Division

1 / 2 = ??

1.0 / 2.0 = ??

1 / 2 = 0

1 and 2 are integer constants.

1.0 / 2.0 = 0.5

1.0 and 2.0 are decimal constants.

Remainder

mod(%) gives you the integer remainder of integer division.

```
out.println(2 % 3);

out.println(3 % 2);
```

OUTPUT

```
2

1
```

Remainder

mod(%) gives you the integer remainder of integer division.

```
num = 45;
out.println(num%10);
out.println(num/10);
```

OUTPUT

```
5

4
```

Remainder

mod(%) gives you the real number remainder of real number division.

```
out.println(9 % 3);

out.println(9.2 % 3);
```

OUTPUT

```
0
0.19
```

divide.java
modulus.java

Precedence

()	HIGH
! ++ --	
* / %	
+ -	
= += -= *= /= %=	
,	LOW

Assignment

```
int num = 10;
out.println(num);

num = num + 5;
out.println(num);

num = 10 * 2 + 7;
out.println(num);
```

OUTPUT

```
10
15
27
```

Assignment

```
num *= 2;
out.println(num);

num /= 5;
out.println(num);

num = num + 4 / 2 - 8;
out.println(num);

num = (4 + 5)/2+7;
out.println(num);
```

OUTPUT

```
54
10
4
11
```

Shortcuts

```
num = 11;
out.println(num);

num++;
out.println(num);

num--;
out.println(num);

num++;
out.println(num);
```

OUTPUT

```
11
12
11
12
```

assignment.java
shortcuts.java

Casting

Casting is used to temporarily change the type of a value.

(int)3.14159
(double)3

Casting is often used to create compatibility among data types.

Casting

```
int one = 0;                    //32 bit int
long big = 453;                 //64 bit int
double dec = 7.56;              //64 bit real

one = dec;                      //illegal
one = big;                      //illegal
one = (int)dec;                 //legal
one = (int)big;                 //legal
```

Casting is often used to create compatibility among data types.

cast.java

Casting

```
int one = 11;
int two = 5;
double dec = (double)one/two;
```

As long as one part of the division is a decimal value, the result will be a decimal.

one is temporarily converted to a double before the division.

Casting

```
out.println("1/2 = " + (1/2));
out.println("(double)1/2 = " + (double)1/2);
out.println("5/2 = " + (5/2));
out.println("5/(double)2 = " + 5/(double)2);
```

OUTPUT

```
1/2 = 0
(double)1/2 = 0.5
5/2 = 2
5/(double)2 = 2.5
```

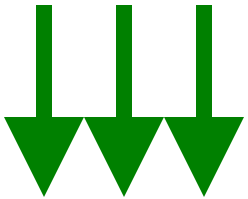
intcast.java

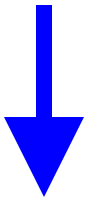
Formatting Numbers

Formatting Output

How to format

What to format





```
out.printf( " %.2f " , 9.237284 );
```

OUTPUT

9.24

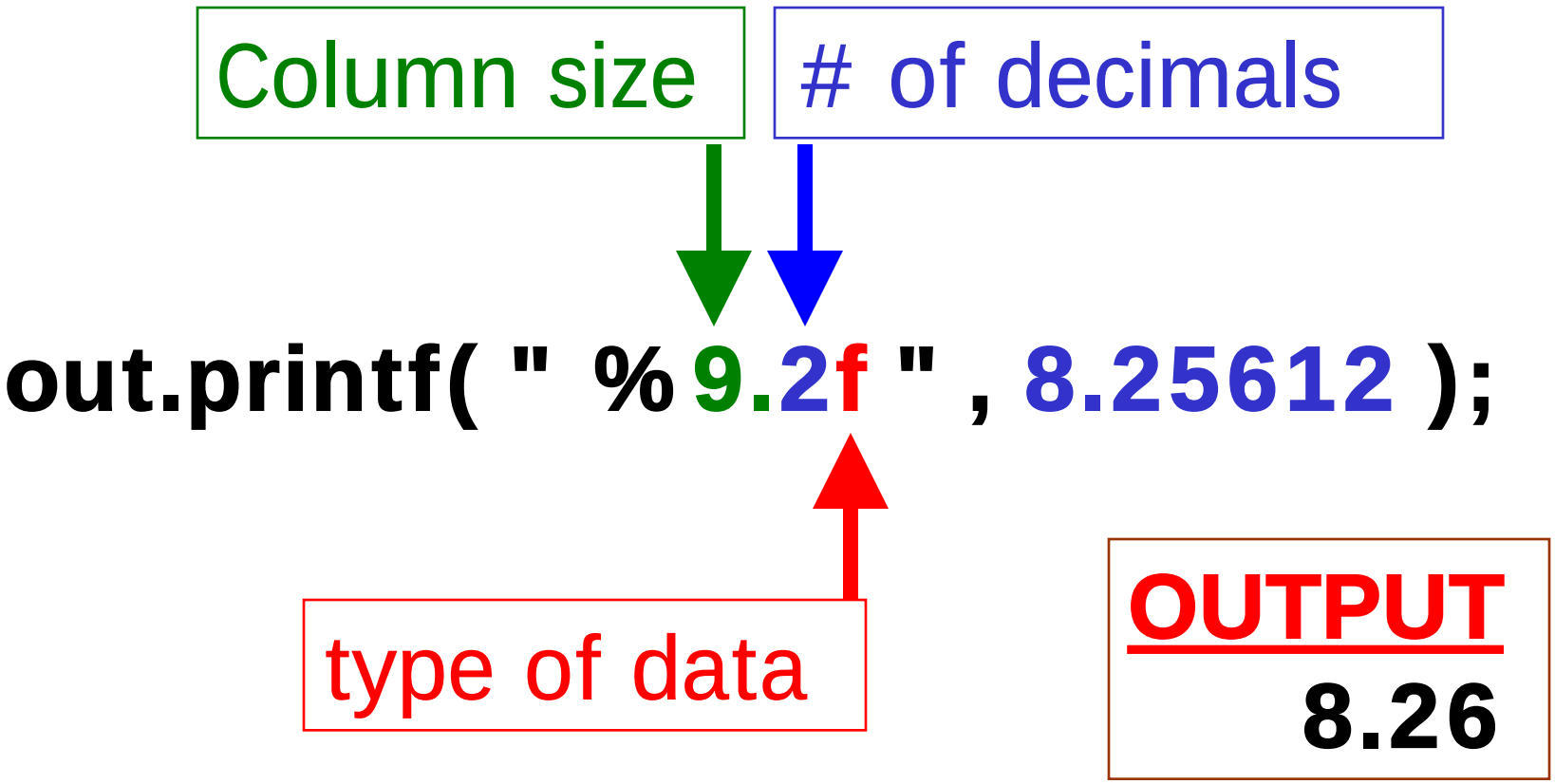
Formatting Output

```
double dec = 9.231482367;
out.printf("dec == %.1f\n",dec);
out.printf("dec == %.2f\n",dec);
out.printf("dec == %.3f\n",dec);
out.printf("dec == %.4f\n",dec);
out.printf("dec == %.5f\n",dec);
```

OUTPUT

```
dec == 9.2
dec == 9.23
dec == 9.231
dec == 9.2315
dec == 9.23148
```

Formatting Output



Formatting Output

```
double dec = 5.3423;
out.println(String.format("%.3f",dec));
out.println(String.format("%12.3f",dec));
out.println(String.format("%-7.3f",dec));
```

OUTPUT

```
5.342
        5.342
5.342   x
```

realformatone.java
realformattwo.java

Formatting Output

```
int num = 923;
out.printf("%d\n", num);
out.printf("%6d\n", num);
out.printf("%-6d\n", num);
out.printf("%06d\n", num);
```

OUTPUT

```
923
   923
923
000923
```

Formatting Output

```
int num = 567;
out.println(String.format("%d",num));
out.println(String.format("%6d",num));
out.println(String.format("%-6d",num));
out.println(String.format("%06d",num));
```

OUTPUT

```
567
   567
567
000567
```

intformatone.java
intformattwo.java

Work on Programs!

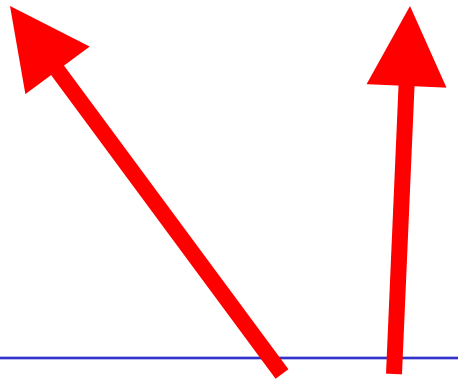
Crank

Some Code!

Defining Parameters

Defining Parameters

```
public void times(  int num1,   int num2 )
{
   out.println(num1*num2);
}
```



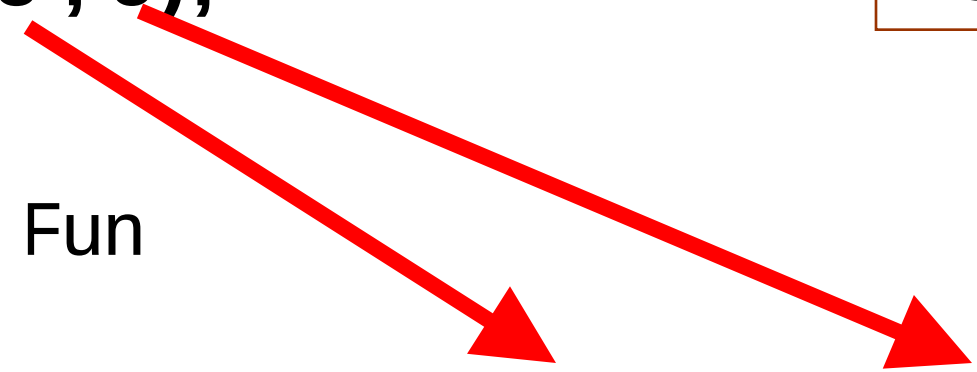
There will be times that we define parameters when we define a method. The parameters allow us to specify the type of data the method will receive.

Passing Parameters

```
//code in main in another class
Fun test = new Fun();
test.times(3 , 5);
```

OUTPUT

15



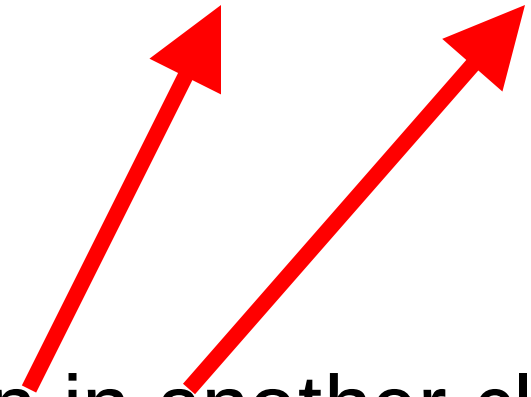
```
public class Fun
{
    public void times(  int num1,   int num2 )
    {
        out.println(num1*num2);
    }
}
```

Passing Parameters

OUTPUT

15

```
public class Fun
{
    public static void times( int one,  int two )
    {
        out.println(one*two);
    }
}
```



```
//code in main in another class
Fun.times(3 , 5);
```

parametersone.java
parameterstwo.java

OOP Pieces

Instance Variables

When you need multiple methods to have access to the same variable, make the variable an instance variable.

The scope of an instance variable is the entire class where the variable is defined.

Instance Variables

```
public class Calc
{
  private int one, two;
  private int answer;


  public void add(){
    answer = one + two;
  }

  public void print(){
    System.out.println(answer);
  }
}
```

Instance variables are shared by all methods in a class.

Private

All members with private access can be accessed or modified only inside the class where they are defined.

Encapsulation

All data members should have private access. A set of public methods should be provided to manipulate the private data.

Modifier Methods

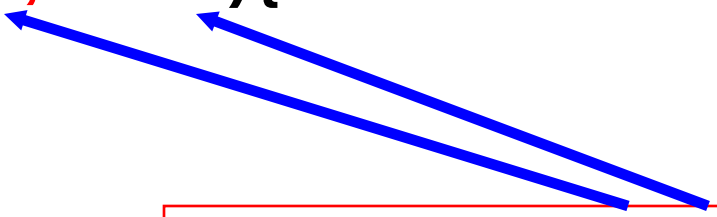
Modifier methods are methods that change the properties of an object.

```
public class Calc
{
  private int one, two;
  private int answer;

  public void setNums( int n1, int n2 ){
    one=n1;
    two=n2;
  }

  public void add(){
    answer = one + two;
  }

  public void print(){
    System.out.println(answer);
  }
}
```



```
test.setNums(4,9);
test.add();
test.print();
```

OUTPUT

13

Modifier Methods

```
public void setSides(int a, int b, int c)
{
    sideA=a;
    sideB=b;
    sideC=c;
}
```

Modifier methods are methods that change the properties of an object.

calc.java
calcrunner.java

triangle.java
trianglerunner.java

Work on Programs!

Crank

Some Code!

A+ Computer Science

MATH / OOP