

CMSC131

Overloading, Ternary Operator,
Switch Statement

Method Overloading

You can "overload" a method name by having multiple implementations of a method with different parameter lists.

Consider these:

```
public int fun1(int i, int j);  
public int fun1(int i, float f);  
public int fun1(float f, int i);  
public int fun1(Double d, Integer i);
```

Method Overloading Limitations

You cannot have overloaded methods differ by only the return type or only by different local names within parameter lists.

These conflict:

```
public int fun1(int i, int j);  
public boolean fun1(int i, int j);
```

These conflict:

```
public int fun1(int i, int j);  
public int fun1(int a, int b);
```

The ternary operator

The ternary operator is of the form

(boolean_expression)?if_true:if_false;

A simple example using assignment

```
String s=(x<0)?"Negative":"Not  
Negative";
```

More useful "tricks" include things such as

```
minVal = (a < b) ? a : b;
```

```
absValue = (a < 0) ? -a : a;
```

The **switch** statement

Basically, a switch means that for simple primitives, instead of:

```
if (option == 1) {code1;}  
else if (option == 2) {code2;}  
else if (option == 3) {code3;}
```

you can use:

```
switch (option) {  
    case 1: code1; break;  
    case 2: code2; break;  
    case 3: code3; break;  
}
```

Copyright © 2012 : Evan Golub