Switch Statement

An alternative to if ... else if ... else

Switch Statement

Selects from 1 of a range of values (int, byte, char or String) and executes the matching code.

Neater than nested “if else” statements, but limited in what it can test.

for example:

if (a>b), if (a<b), if (a==b) if (!a==b)

Switch only tests for **equality**

Switch Statement

```
switch (int){

case 1 : do something, break;

case 2 : do something, break;

…..

case n: do something, break;

default: no match, break;

}
```

Purpose of word break in a switch statement

Break signifies the end of a specific case

You *must* use break, otherwise all subsequent code will be also executed, until the next break is met

However there maybe occasions where you wish this to happen

Example of where break would be left out of some cases – A vowel counter

```
// Input taken into char variable charA

switch (charA){

case a :  ; // If charA is either a, e, I or o, it will

case e: ; // fall through until it hits the u case

case i : ; // at which point vowelCount is incremented

case o : ;

case u: vowelCount++, break;

default: consonantCount++, break;}
```

Default Case

- The default, as the name suggests, will be executed if no other case is selected
- Usually left til last BUT not necessarily so
  - A favourite exam question

Behaviour of a Switch

A switch is executed in the following matter

1. Evaluates the variable being tested against the entire set of cases
2. Goes to the most relevant case and executes the code
3. Continues to execute until it hits the next break;
    - So watch out for “FALL THROUGH”

Links

- http://docs.oracle.com/javase/tutorial/java/nutsandbolts/switch.html
- http://www.leepoint.net/notes-java/flow/switch/switch-general.html