Lecture # 9

The `switch` statement

Introduction to Programming

The `switch` statement

The `if-else-if` control structure can be used to test for multiple cases.

The `if` statement can become cumbersome if testing for multiple, specific conditions.

Java provides a special structure for this purpose. The **`switch` control structure** is used when a variable is to be tested against multiple cases.

switch example

```
int choice;
:     //rest of code here
:
switch(choice)
{
   case 1: System.out.print("option 1 chosen");
           break;
   case 2: System.out.print("option 2 chosen");
           break;    //If this case executes switch ends here
    case 3: System.out.print("option 3 chosen");;
           break;
   default: System.out.print("Invalid option!");
} //end switch
```

The `switch` statement

The `switch` statement consists of
- a `switch` expression,
- a series of `case` clauses containing a constant, and
- an optional `default` clause.

A `switch` statement may be used when

- a variable (`int` or `char`) is being checked for an exact match;
- the check involves specific values of that variable, for example 'A', 'B‘, 10, 20, and not ranges like >39
- Java 7 allows use of `String` in `switch` statement

The `switch` statement

```
switch(variable)
{
    case constant1:  statement(s);
                     break;
    case constant2: // Sharing Cases
    case constant3:  statement(s);
                     break;
    case constant4:  statement(s);
                     break;
    default:    statement(s);
}
```

There can be any number of `case` clauses.

The `switch` statement

```
switch(grade)
{
      case 'A' : System.out.print("Excellent Student");
                 break;
      case 'B' : System.out.print("Good Student");
                 break;
      case 'C' :  // Sharing Cases
      case 'D' : System.out.print("OK Student");
                 break;
      case 'E' :  // Sharing Cases
      case 'F' : System.out.print("Weak Student");
                 break;
      default : System.out.print("Invalid grade entered");
}
```

Ensure the `switch` statement is indented appropriately

In the above example, the variable `grade` will be evaluated

- if `grade` matches any of the `case` constants then the statement(s) in that `case` clause will be executed. Execution will continue until <u>either</u> a `break` is encountered <u>or</u> the end of the `switch` statement is encountered.

- if `grade` does not match any of the `case` constants then the `default` statement(s) will be executed.

How a `switch` statement operates

- Evaluate the `switch` variable.

- Check each `case` constant for a match to the `switch` variable and execute the statements until `break` encountered
- If there is no `case` that matches the `switch` expression go to the `default`
- If there is no `default` terminate the `switch` statement.

- Terminate the `switch` statement when either a `break` is encountered <u>or</u> the end of the `switch` statement is encountered.

`default`

- The `default` is optional (like the `else` in an `if/else`). If it is present however, it should be placed last.
- The `default` statement is equivalent to the `else` part of the `if...else` and `if...else...if` statements – it catches situations where none of the conditions are met.

`break;`

- Every branch of the switch should be terminated by a `break` statement.
- The `break` is not mandatory, however, if the `break;` is not present, execution *falls through* to next branch, until end of switch is reached.
  - without the `break` the cases following the match will also be executed - it will not `break` out of the statement when required.

Multiple cases

Several `case`s can share the same set of statements as shown:

```
switch(grade)
{
  case 'A': // Sharing Cases
  case 'a': System.out.print("70 - 100");
            break;
       .....
  default: System.out.print("Invalid grade
  entered");
}//end switch
```

Multiple cases

```
switch(grade)
{
      case 'A':
      case 'a':
             System.out.print("Excellent Student");
             break;
      case 'B':
      case 'b':
             System.out.print("Good Student");
             break;
      case 'C': case 'c':
      case 'D': case 'd':
             System.out.print("OK Student");
             break;
      case 'E': case 'e':   // Sharing Cases
      case 'F': case 'f':
             System.out.print("Weak Student");
            break;
      default: System.out.print("Invalid grade entered");
}//end switch
```

Limitations of `switch`

- The `switch` is limited
  - can only be used with variables of type `int` or `char` and `String` objects
  - Only **one** variable can be checked in a `switch` statement
  - The values in case clause must be constants
  - Can only be used to test for an exact match.
    - It cannot be used to test if value lies in a range

```
case: < 2;      // syntax error
```

- Any `switch` statement can be rewritten as an `if-else-if`, but not every `if-else-if` can be rewritten as a `switch`.

Rewrite using `switch`

```
if(marriageStatus == 'S')
{
   System.out.print("Single");
}
else if(marriageStatus == 'M')
{
   System.out.print ("Married");
}
else if (marriageStatus == 'W')
{
   System.out.print("Widowed");
}
else
{
   System.out.print("Invalid code");
}
```

```
switch(marriageStatus)
{
case 'S':
       System.out.print("Single");
       break;
case 'M':
       System.out.print("Married");
       break;
case 'W':
       System.out.print("Widowed");
       break;
default:
       System.out.print("Invalid code");
}
```

Rewrite using `switch`

```
if(marriageStatus == 'S' || marriageStatus == 's'))
{
   System.out.print("Single");
}
else if(marriageStatus == 'M' || marriageStatus == 'm'))
{
   System.out.print ("Married");
}
else if (marriageStatus == 'W' || marriageStatus == 'w'))
{
   System.out.print("Widowed");
}
else
{
   System.out.print("Invalid code");
}
```

```
switch(marriageStatus)
{
   case 'S':
   case 's':
            System.out.print("Single");
            break;
   case 'M':
   case 'm':
            System.out.print("Married");
            break;
   case 'W':
   case 'w':
            System.out.print("Widowed");
            break;
   default:System.out.print("Invalid code");
}
```