CONDITIONAL STATEMENTS

THE IF FAMILY – 1/2

- If conditional expression
  - True → then execute one set of statements
  - False → finds the end of the if block and executes the first statement following the block

- Syntax

```
if (conditional expression)
{
        //statements
}
```

THE IF FAMILY – 2/2

- The if family branching statements are
  - if else
  - if elsif else

- They follow zero, one or many rule
  - Only one if statement is allowed with conditional expression
  - An if clause may have zero or one else blocks
  - An if clause may have zero, one or many elsif clauses

THE IF ELSE STATEMENT

- Syntax

```
if (cond exp)
{
      block of stmts
}
else
{
      block of stmts
}
```

- The **else** clause is executed only when **if** statement is false
- An **if** statement can exists without **else** block, but **else** cannot exists without **if**
- An **if** clause may have only one **else** block

THE IF ELSIF ELSE STMT – 1/2

- Allows to have conditional expression after a false result from if clause
- The elsif must come before else clause if else clause is associated with the if statement
- Can have many elsif clauses associated with if clause as desired

THE IF ELSIF ELSE STMT – 2/2

- Syntax

```
if (cond exp)
{
        stmts
}
elsif (cond exp)
{
        stmts
}
optional additional elsif clause
optional else clause
```

NESTED IF CLAUSES VS ELSIF CLAUSES – 1/3

- Both can execute only if the previous if condition if false

- Primary difference occurs when the else if clause completes and code starts to exit each enclosing nested if clause

- As nested else if clause is completed, code branches to end of its enclosing scope, allowing additional statements to execute as each if else clause completes its block of statements

NESTED IF CLAUSES VS ELSIF CLAUSES – 2/3

```
if (cond)
{
 stmts
}
else
{
        if (cond)
        {
              stmts
        }
        stmts
}
```

- **All statements inside else block is executed whether or not condition is true**

NESTED IF CLAUSES VS ELSIF CLAUSES – 3/3

- With the elsif clause, only statements inside elsif block are executed

```
if (cond)
{
        stmt
}
elsif (cond)
{
        stmt
}
else
{
        stmt
}
```

SWITCH SIMULATION – 1/4

- **Switch statement branches to a single block of statements among many statements**
- **Once block of statements finish execution, code branches to next switch statement**
- Two **reasons** that switch statement is favored
  - Can change long list of if else if or if elsif clauses into clearly separated logical groups
  - Easy to change conditional expression that control each switch block

SWITCH SIMULATION – 2/4

- **Perl does not have built in switch statement,** we have to simulate one

```
SWITCH:
{
        if (cond1)
        {
                last SWITCH;
        }
        if (cond2)
        {
                last SWITCH;
        }
        DEFAULT:
        {
                last SWITCH;
        }
}
```

SWITCH SIMULATION – 3/4

- SWITCH simulation includes
  - a **SWITCH** label
  - **Conditional expression** as required
  - **DEFAULT** label that executes in the event when all condition evaluates to false

SWITCH SIMULATION – 4/4

- Ending statement of every SWITCH block of statements is the ***last*** command
- It causes entire SWITCH block of statements to be exited
- Any statement in SWITCH block following the last statement will not be executed

LABELS - 1/3

- It is marker for Perl interpreter
- **It identifies a particular line in the code**
- Perl statements may then refer to that line by the label's name
- **A label is usually to mark the beginning of new block of statements**

LABELS - 2/3

- Rules for Perl labels
  - A label can go on **any line**
  - Must **begin with char** followed by any combination of letters or numbers that end with colon. **Ending colon is required**
  - **Cannot use reserved** words
  - **Case sensitive.** By convention, they are upper case for easy recognition

LABELS - 3/3

- The **last** command exits whatever block of statements is referenced by its label

```
    last LABEL;
```

- **Label is not required with the last command**

- Can also use last command like

```
    last;
```

- **When used w/o label, last exits its block statements**

TERNARY OPERATOR – 1/2

- **Replaces entire 'if else' construct**

```
(cond exp) ? True exp : false exp
```

TERNARY OPERATOR – 2/2

```
if ($cmax < $ no)
{
      $max = $no;
}
else
{
      $max = $cmax;
}
```

Use the following code below:

```
$max = ($cmax < $no) ? $no : $cmax;
```