Data Structures and Program Design in C

Topic 3 : Control Flow

Dr Manjusri Wickramasinghe

Outline

- What are statements and blocks?
- Selecting Constructs
  - `if- else` Statement
  - `switch` Statement
  - Ternary Operator
- Looping Constructs
  - `while` Loop
  - `do – while` Loop
  - `for` Loop
- Jump Statements

What are Statements and Blocks?

- **Statement:-** is a line of code written in a programming language commanding to achieve a specific task.
  - Statements are terminated by a semi-colon (**;**) in the C programming language.

- **Block:-** Is a set of statements or declarations grouped together so that when it executes, it is syntactically equivalent to a single statement.
  - Blocks in C language are determined by the curly braces (e.g. { … }).
  - No terminating character as in statements.

if-else …(1)

- The if-else statement is used to make decisions in the program flow.

```
if(<expression>)
{
      //code when the expression is true
}
[else
{
      //code when the expression is false
}]
```

`if-else` …(2)

- The else part of the if-else statement is optional, and it is correct only to have an if statement.

- The expression is evaluated to be true when the expression yields a Boolean outcome of true or has a **non-zero value**.
  - When true, the code in if-block will be executed.

- When the expression is evaluated to be false when the expression yields a Boolean outcome of false or has a **zero value**.
  - When false the code in the else-block will be executed *if an else block is present*.

`if-else` …(3)

- Though it is possible to have if-else statements without braces, **it is a good programming practice to always put braces around such statements and to indent them properly**.

- Multiple if-else statements can be stacked to create a multiway decision.
  - Also called the if-else ladder.

- Four (4) variants as `if`, `if-else`, `nested if-else` and `if-else ladder`.

`if-else` …(4)

- General Use
  - Allows decision-making on a specific condition.
  - Allows complex decision-making by combining multiple if-else statements

- Possible Problems
  - The readability of the code can become a problem with multiple statements being nestled together.
  - Code duplication in places where the same logic is repeated.

switch …(1)

- The switch statement is a multi-way decision that tests whether an expression matches one of the defined **constant integer values**.

```
switch(<expression>)
{
    case constant-expression: statements
    case constant-expression: statements
    default: statements
}
```

`switch` …(2)

- If the expression matches a case, the execution starts from that case and **falls through**.
- If no cases match, the `default` case is executed.
- Default case is optional.

Jump Statements*

- The C Programming Language has four (4) types of jump statements
  - break
  - continue
  - goto
  - return

* → The four types will be covered under separate topics when the usage is required.

`break` statement

- The `break` statement terminates the **immediately enclosing** loop block or `switch` block.

- The control passes to the statement immediately after the enclosing block.

- If there is no enclosing block, the compiler generates an error.

Ternary Operator ...(1)

- Ternary operator is a conditional operator in the C programming language.

- Operator is represented by `?:`

```
<condition> ? <if-true> : <if-false>
```

Ternary Operator ...(2)

- Ternary operator is not a replacement for the `if-else` statement.

- If improperly used, the ternary operator would create readability issues in the code.

Iterative Statements

- Iterative statements are also called loops.

- There are three (03) loops in the C Programming Language.
  - while loop
  - for loop
  - do – while loop

- All loops have three (03) components. That is
  - Loop Condition
  - Loop Counter
  - Loop Body

while Loop

```
while(<expression>)
{
    //code when expression
    //is true
}
```

`for` Loop

```
for(<initialization>; <condition>; <counter>)
{
    // loop body
}
```

do-while Loop

```
do
{
    //loop body
} while(<expression>);
```

Loops

- Types of loops
  - Entry controlled loops
  - Exit controlled loops

Jump Statements*

- The C Programming Language has four (4) types of jump statements
  - break → terminates the immediately enclosing block.
  - continue
  - goto
  - return

* → The four types will be covered under separate topics when the usage is required.

`continue` Statement

- Is a jump statement that takes the program control to the next iteration of the loop.

- The code beyond the `continue` statement is not executed and will be skipped.

`goto` Statement

- This is an explicit jump.

```
goto <label-name>;
```

- Label are defined using the following syntax.

```
<label-now:>
```

Jump Statements*

- The C Programming Language has four (4) types of jump statements
  - break → terminates the immediately enclosing block.
  - continue → skips the remainder of the loop
  - goto → explicit jump to label.
  - return

* → The four types will be covered under separate topics when the usage is required.

Questions?