C Programming

Trainer: Ketan Kore

Control Statements

- Decision or Selection
  - if-else
  - switch-case
- Iteration (loop)
  - for
  - while
  - do-while
- Jump
  - break
  - continue
  - goto
  - return

Ternary/conditional operator

```
if  (condition) {
        //  execute if  condition is true
}
else {
        //  execute if  condition is false
}
```

- if-else can be nested within each other.

condition ? expression1 : expression2

- If condition is true, expression1 is executed; otherwise expression2 is executed.
- Ternary operators can also be nested.
- expression1 & expression2 must be expressions (not statement).
  - expression – evaluate to some value.
  - statement – C statement ends with ;

switch-case

```
switch (expression) {
     case const-expr1:
            statement(s);
            break;
    case const-expr2:
            statement(s);
            break;
    ...
    default:
            statement(s);
            break;
}
```

- Switch-case is used to select one of the several paths to execute depending on value of int expression.
- case constants cannot be duplicated.
- break statement skips remaining statements and continues execution at the end of switch closing brace.
- If break is missing, statements under sub-sequent case continue to execute.
- default case is optional and it is executed only if int expression is not matching with any of the case constant.
- Sequence of cases and default case doesn't matter.

Loops

- Control statements used for repeating a set of instructions number of times is called as “LOOP”.
- Every loop has
    - Initialization statement
    - Terminating condition
    - Modification statement(Increment/Decrement)
    - Body of loop
- The variable that is used for terminating condition is referred as ‘loop variable’.

while loop

- Used to repeat a statement (or block) while an expression is true (not zero).

- Syntax:

```
initialization;
while(condition) {
        statement1;
        statement2;
        modification;
}
```

for loop

- Used to repeat a statement (or block) while an expression is true (not zero).
- Syntax:

```
for(initialization; condition; modification) {
        statement1;
        statement2;
}
```

Thank you!