Online lecture notes
Paper Name: Programming Language 'C'
Topic: Selection Construct

Date: 07/02/2023
By: Ashutosh Pawan
Department of BCA
H. D. Jain College, Ara

Selection Constructs available in 'C' are as:

1. **if Selection construct**

Multiple statements

```
if(<condition>)
{
<statements>;
/*these statements are executed when
condition becomes true*/
-------------;
}
```

Single statement

```
if(<condition>)
<statement/True_statement>;
/*There is no need to use { and } if only one
task is to be carried out*/
/*Statement that gets executed when
condition becomes true is called
True_statement*/
```

2. **if – else Selection construct**

Multiple statements

```
if(<condition>)
{
<statements>;
/*these statements are executed when
condition becomes true*/
-------------;
}
else
{
<statements>;
/*these statements are executed when
the condition becomes false*/
-------------;
}
```

Single statement

```
if(<condition>)
<statement/True_statement>;
else
<statement/False_statement>;
/*There is no need to use { and } if only one
task is to be carried out*/
/*Statement that gets executed when
condition becomes true is called
True_statement* and those statement that
gets executed when condition becomes
false is called false statement*/
```

3. **if – else ladder selection construct**

Multiple statements

```
if(<condition1>)
{
<statements>;
/*these statements are executed when
condition1 becomes true*/
-------------;
}
else if(<condition2>)
{
<statement>;
/*this gets executed when the
condtion2 becomes true but remember
condition2 is checked when the
condition1 becomes false*/
--------------------;
}
```

Single statement

```
if(<condition1>)
<statement/True_statement_condition1>;
else if(<condition2>)
<statement/True_statement_condition2>;
else if(<conditionN>)
<statement/True_statement_conditionN>;
else
<False_statement>;
/*There is no need to use { and } if only one
task is to be carried out*/
/*Statement that gets executed when
condition becomes true is called
True_statement*/
```

```
else if(<conditionN>)
{
<statement>;
/*this gets executed when the
condtionN becomes true but remember
conditionN is checked when the
conditionN-1 becomes false*/
-------------------;
}
else
{
<False_statement>;
/*this gets executed when all above
condition becomes false*/
-----------------------;
}
```

4. **Nested if – else selection construct**

Multiple statements

```
if(<Outer_condition>)
{
<statements>;
/*these statements are executed when
Outer_condition becomes true*/
------------;
if(<Inner_condition>)
{
<statements>;
/*these statements are executed when
Inner_condition becomes true*/
------------;
}
else
{
<statements>;
/*these statements are executed when
the Inner_condition becomes false*/
------------;
}
------------;
}
else
{
<statements>;
/*these statements are executed when
the Outer_condition becomes false*/
------------;
if(<Inner_condition>)
{
<statements>;
/*these statements are executed when
Inner_condition becomes true*/
```

Single statement

```
if(<Outer_condition>)
if(<Inner_condition>)
<statement/True_statement>;
else
if(<Inner_condition>)
<statement/True_statement>;
else
<statement/False_statement>;
/*There is no need to use { and } if only
one task is to be carried out*/
/*Statement that gets executed when
condition becomes true is called
True_statement* and those statement that
gets executed when condition becomes
false is called false statement*/
```

```
-------------;
}
else
{
<statements>;
/*these statements are executed when
the Inner_condition becomes false*/
-------------;
}
-------------;
}
```

5. **switch – case selection construct**

```
switch(<Variable_Name>)
{
case  <value1>: <statements_value1matched>;
                        break;
case  <value2>: <statements_value2matched>;
                        break;
--------------------------------------------------------;
case  <valueN>: <statements_valueNmatched>;
                        break;
default : <default_statement>;
                        break;
}
/*when no case statement gets executed, default case statement gets executed*/
```

Note: The pages have been divided into two columns vertically to show two different syntaxes of the selection construct. In first vertical part the given syntax can be used when the programmer wants to perform multiple tasks and the syntax given in second vertical part can be used when the programmer wants to perform a single task.