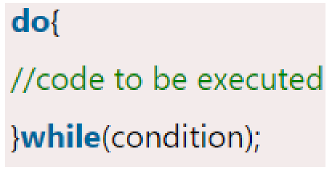
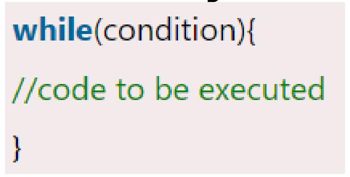
C Loops

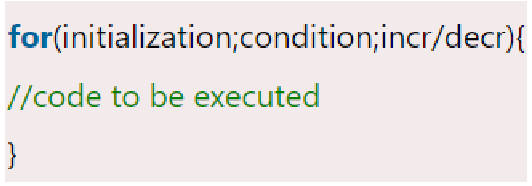
- Loops can execute a block of code until a specific condition is satisfied.
- Loops are handy because they save time, reduce errors, provides code re-usability and they make code more readable.
- Types of C Loops:
  - do while – It is similar to while loop, except that it tests the condition at the end of the loop body.
  - while - It tests the condition before executing the loop body.
  - for - It is used when you know exactly how many iterations are needed.



```
do{
//code to be executed
}while(condition);
```



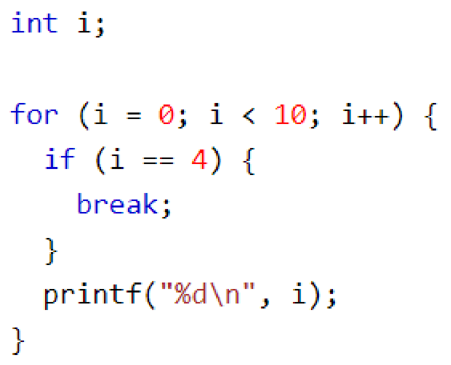
```
while(condition){
//code to be executed
}
```



```
for(initialization;condition;incr/decr){
//code to be executed
}
```

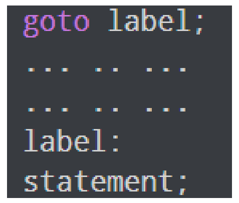
C Loop Control Statements

- Loop control statements change execution from its normal sequence. When execution leaves a scope, all local variables that were created in that scope are destroyed.
- C supports following control statements:
  - break – Terminates the loop/switch and transfers execution to the statement immediately following the loop or switch.
  - continue - It skips the current iteration of the loop and continues with the next iteration
  - goto – It transfers the control to the labeled statement.



```c
int i;

for (i = 0; i < 10; i++) {
  if (i == 4) {
    break;
  }
  printf("%d\n", i);
}
```



```c
goto label;
... .. ...
... .. ...
label:
statement;
```