Decision making and looping

CHAPTER 29

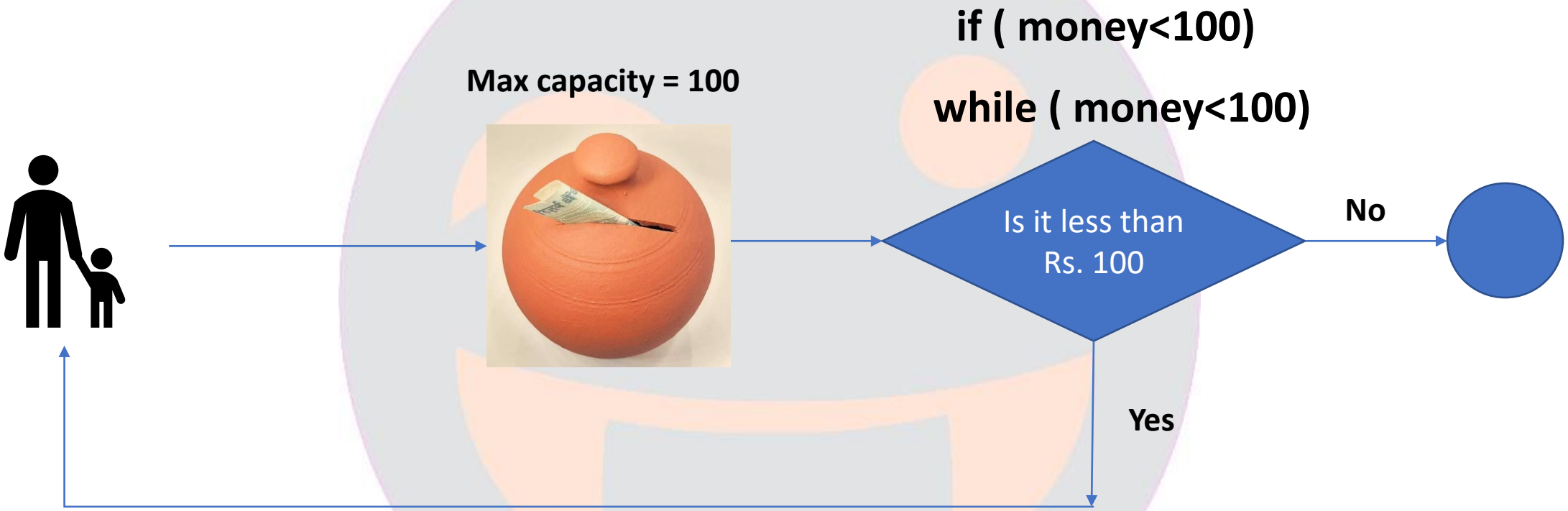
SURESH TECHS

C PROGRAMMING COURSE

Loop

Repeating the same process multiple times until it meets a specific condition

Save money in the kiddy bank



Write a program to find the number of days to save minimum Rs.100

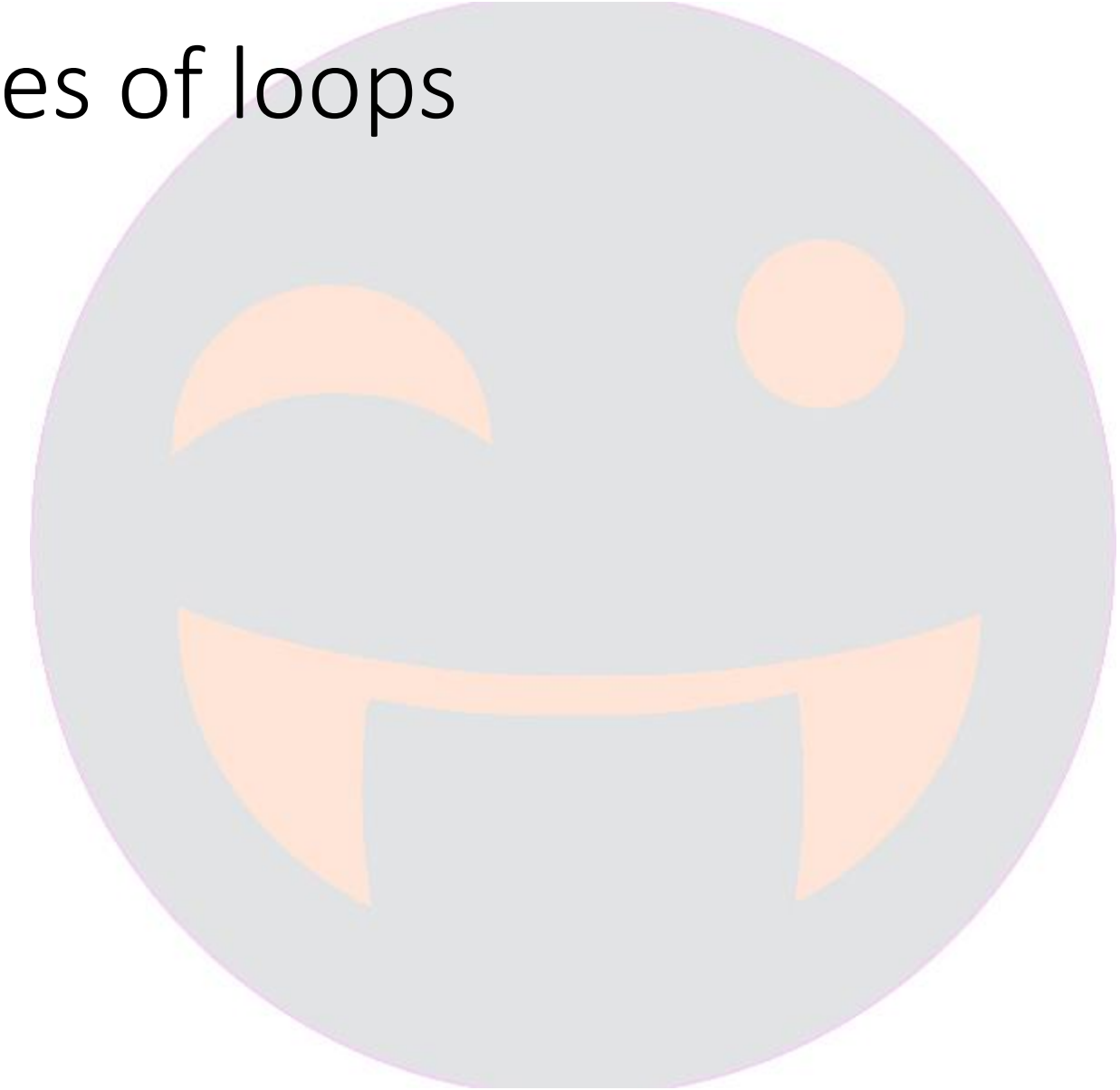
Count number of days to save Rs.100

```c
#include<stdio.h>
int main(){
    int money = 0;
    int fatherGave;
    int days = 0;
    while(money<100){
        printf("How much father gave? ");
        scanf("%d",&fatherGave);
        money = money+fatherGave;
        days = days+1;
    }
    printf("Total days to save minimum Rs 100: %d\n",days);
    printf("Total money saved: %d",money);

    return 0;
}
```

Three types of loops

- while
- do while
- for



while

```
while (condition) {
statements;
}
```

True – non zero value
False - 0

Print all even numbers till n(provided by user)

```c
#include<stdio.h>
int main(){
    int n;
    int number=0;
    printf("Enter a number and I will display all even numbers till that number starting from 0: ");
    scanf("%d",&n);
    while(number<n){
        if(number%2==0){
            printf("%d\n",number);
        }
        number++;
    }

    return 0;
}
```

```c
#include<stdio.h>
#include<math.h>
int main(){
    int n;
    int number= -INFINITY;
    printf("Enter a number and I will display all even numbers till that number
starting from 0: ");
    scanf("%d",&n);
    while(number<n){
        if(number%2==0){
            printf("%d\n",number);
        }
        number++;
    }

    return 0;
}
```

do while

```
do {
statements
} while (condition);
```

1. Statements in the loop **executes at least once**
2. Condition is checked **at the end(exit)**
3. That is why **do while** is called **exit controlled loop**

1. while is called **entry controlled loop**

The while loop is executed **only when the condition is true**, but sometimes the statement needs to be **executed at least once**, so for that **do-while loop has to be used**

do while – Guess output of the programs?

```c
#include<stdio.h>
int main(){
    int n;
    //print numbers in the descending order till 0
    printf("Enter a number: ");
    scanf("%d",&n);
    do{
    printf("%d\n",n);
    n--;
    }while(n>0);

    return 0;
}
```

```c
#include<stdio.h>
int main(){
    int n;
    //print numbers in the descending order till 0
    printf("Enter a number: ");
    scanf("%d",&n);
    while(n>0){
        printf("%d\n",n);
        n--;
    }

    return 0;
}
```

while() loop	do...while() loop
It is known as an ***entry-controlled*** loop.	It is known as an ***exit-controlled*** loop.
In a while loop, first we check the condition, if it is true, the control will go inside the loop.	In a do...while() loop, we are entering the body of the loop without checking any condition.
we use a **while keyword**.	we use a **do-while keyword**.

Program to print numbers from 1 to n

```c
#include<stdio.h>
int main(){
    int n;
    int currentValue = 1;
    //print numbers starting from 1 till n
    printf("Enter a number: ");
    scanf("%d",&n);
    while(currentValue<=n){
        printf("%d\n",currentValue);
        currentValue++;
    }
    return 0;
}
```

for

- Very powerful
- Completely different than while and do while

```
for (initializationStatement; conditionTest; updateStatement) {
        //Statements to be executed
}
```

```
for (initializationStatement; conditionTest; updateStatement) {
        //Statements to be executed
}
```

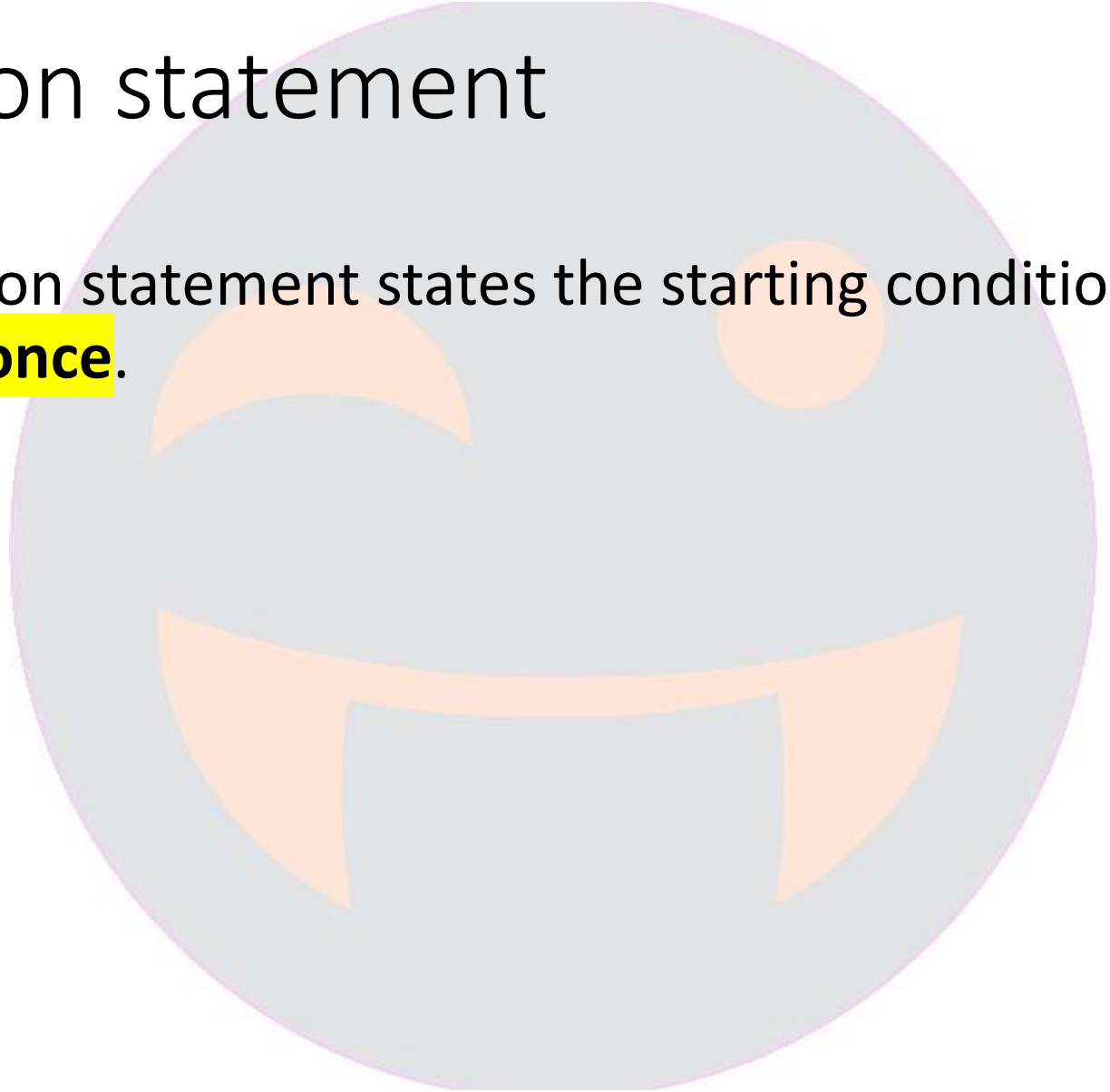
```c
#include<stdio.h>
int main(){
    int n;
    int currentValue = 1;
    //print numbers starting from 1 till n
    printf("Enter a number: ");
    scanf("%d",&n);
    while(currentValue<=n){
        printf("%d\n",currentValue);
        currentValue++;
    }
    return 0;
}
```

```
for (initializationStatement; conditionTest; updateStatement) {
        //Statements to be executed
}
```

```c
#include<stdio.h>
int main(){
    int n;
    //print numbers starting from 1 till n
    printf("Enter a number: ");
    scanf("%d",&n);
    for(int currentValue=1;currentValue<=n;currentValue++){
       printf("%d\n",currentValue);
    }
    return 0;
}
```

Initialization statement

- The initialization statement states the starting condition for the loop. **It is run only once**.



Condition statement

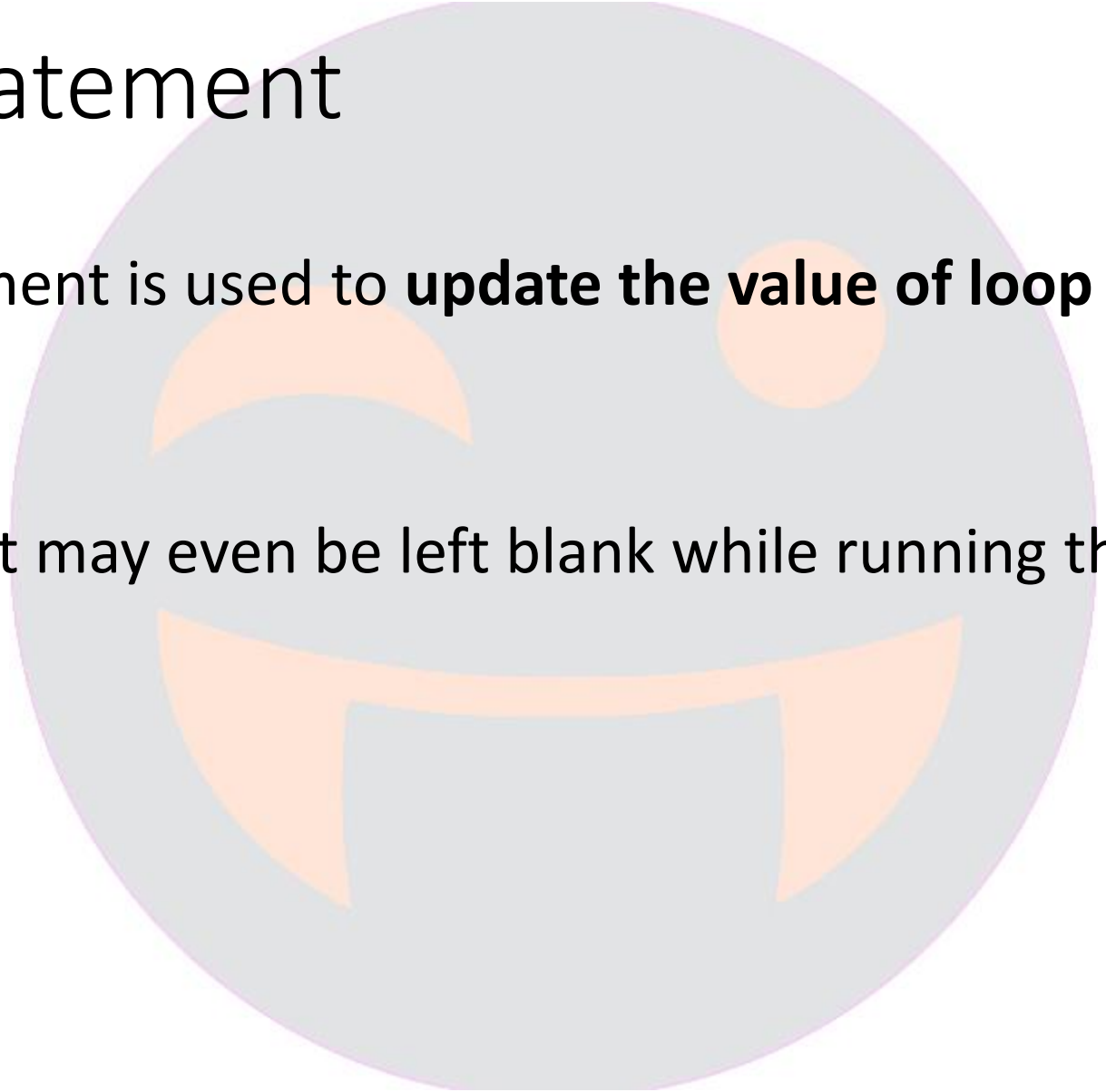
- The condition statement is used **to control the flow of execution of the loop** based on some conditions.

- If this statement is not declared properly, it may lead to an **infinite loop**.

Update statement

- Update statement is used to **update the value of loop control variables**.

- This statement may even be left blank while running the loop.



Program to print odd numbers till n

```c
#include<stdio.h>
int main(){
    int n;
    printf("Enter a number: ");
    scanf("%d",&n);
    for(int i=0;i<=n;i++){
            if(i%2!=0){
                printf("%d\n",i);
            }
    }
    return 0;
}
```

No initialization

```c
#include<stdio.h>
int main(){
    int n;
    printf("Enter a number: ");
    scanf("%d",&n);
    for(int i=0;i<=n;i++){
            if(i%2!=0){
                printf("%d\n",i);
            }
    }
    return 0;
}
```

```c
#include<stdio.h>
int main(){
    int n;
    int i;
    printf("Enter a number: ");
    scanf("%d",&n);
    for(;i<=n;i++){
            if(i%2!=0){
                printf("%d\n",i);
            }
    }
    return 0;
}
```

No updation

```c
#include<stdio.h>
int main(){
    int n;
    int i;
    printf("Enter a number: ");
    scanf("%d",&n);
    for(;i<=n;i++){
            if(i%2!=0){
                printf("%d\n",i);
            }
    }
    return 0;
}
```

```c
#include<stdio.h>
int main(){
    int n;
    int i;
    printf("Enter a number: ");
    scanf("%d",&n);
    for(;i<=n;){
            if(i%2!=0){
                printf("%d\n",i);
            }
            i++;
    }
    return 0;
}
```

Same as while

Differences between for and while loop

for loop runs a known number of times, whereas a **while loop** runs **until a condition is met**.

This means that **we may use for loops if we know how many times to iterate, and a while loop when we don't**.

No initialization, condition and updation

```c
#include<stdio.h>
int main(){
    for(;;){
        printf("Subscribe to suresh techs channel");
    }
    return 0;
}
```

Write a program for n^{th} table

```c
#include<stdio.h>
int main(){
printf("5 * 1 = 5\n");
printf("5 * 2 = 10\n");
printf("5 * 3 = 15\n");
printf("5 * 4 = 20\n");
printf("5 * 5 = 25\n");
printf("5 * 6 = 30\n");
printf("5 * 7 = 35\n");
printf("5 * 8 = 40\n");
printf("5 * 9 = 45\n");
printf("5 * 10 =50\n");
return 0;
}
```

```
5 * 1 = 5
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
5 * 6 = 30
5 * 7 = 35
5 * 8 = 40
5 * 9 = 45
5 * 10 =50
```

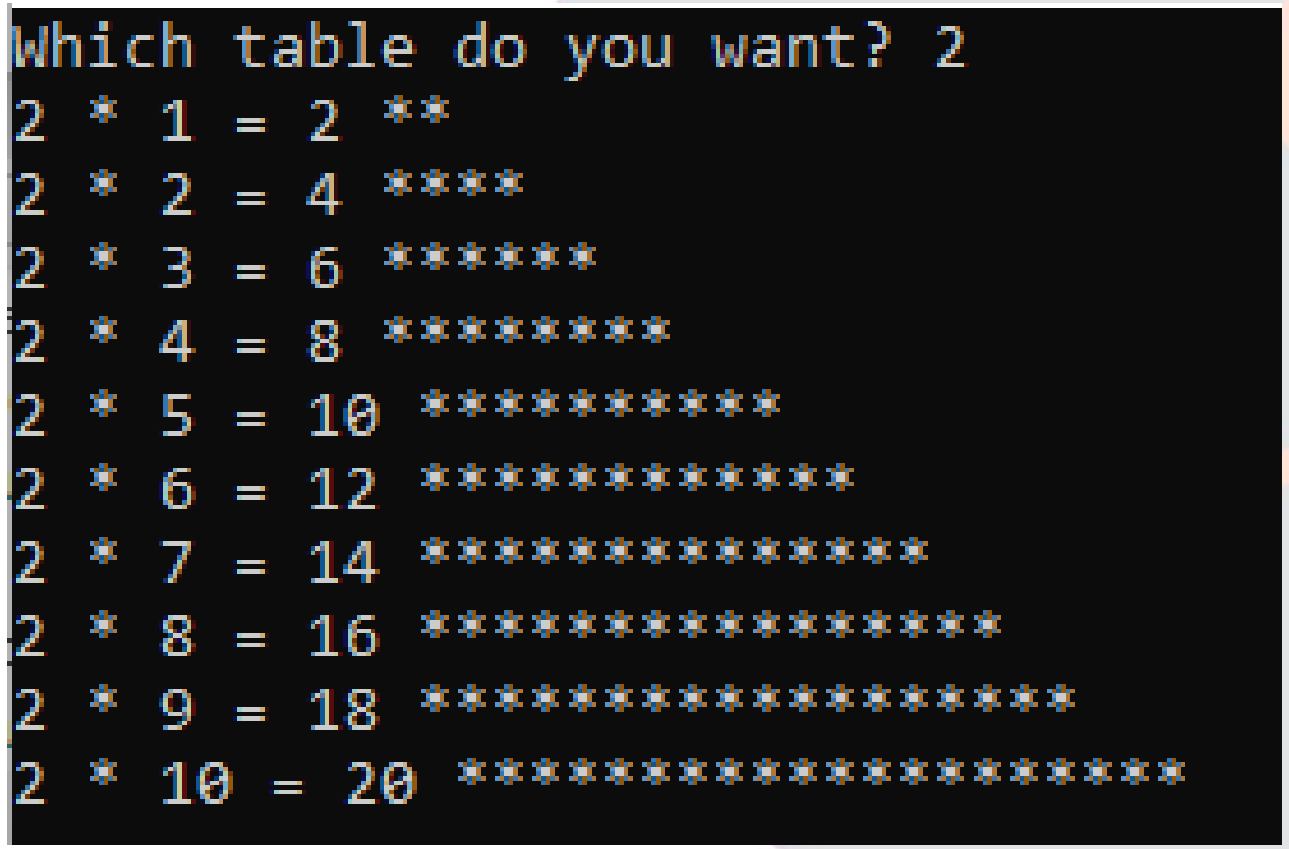
Write a program for n^{th} table

```c
#include<stdio.h>
int main(){
    int n;
    printf("Which table do you want? ");
    scanf("%d",&n);
    for(int i=1;i<=10;i++){
        printf("%d * %d = %d\n",n,i,n*i);
    }
    return 0;
}
```

Loops inside loops? – Nested loops

```
for(initialization; condition; increment/decrement)
{
for(initialization; condition; increment/decrement)
{
statement ;
}
}
```

Print number of stars of the value of table

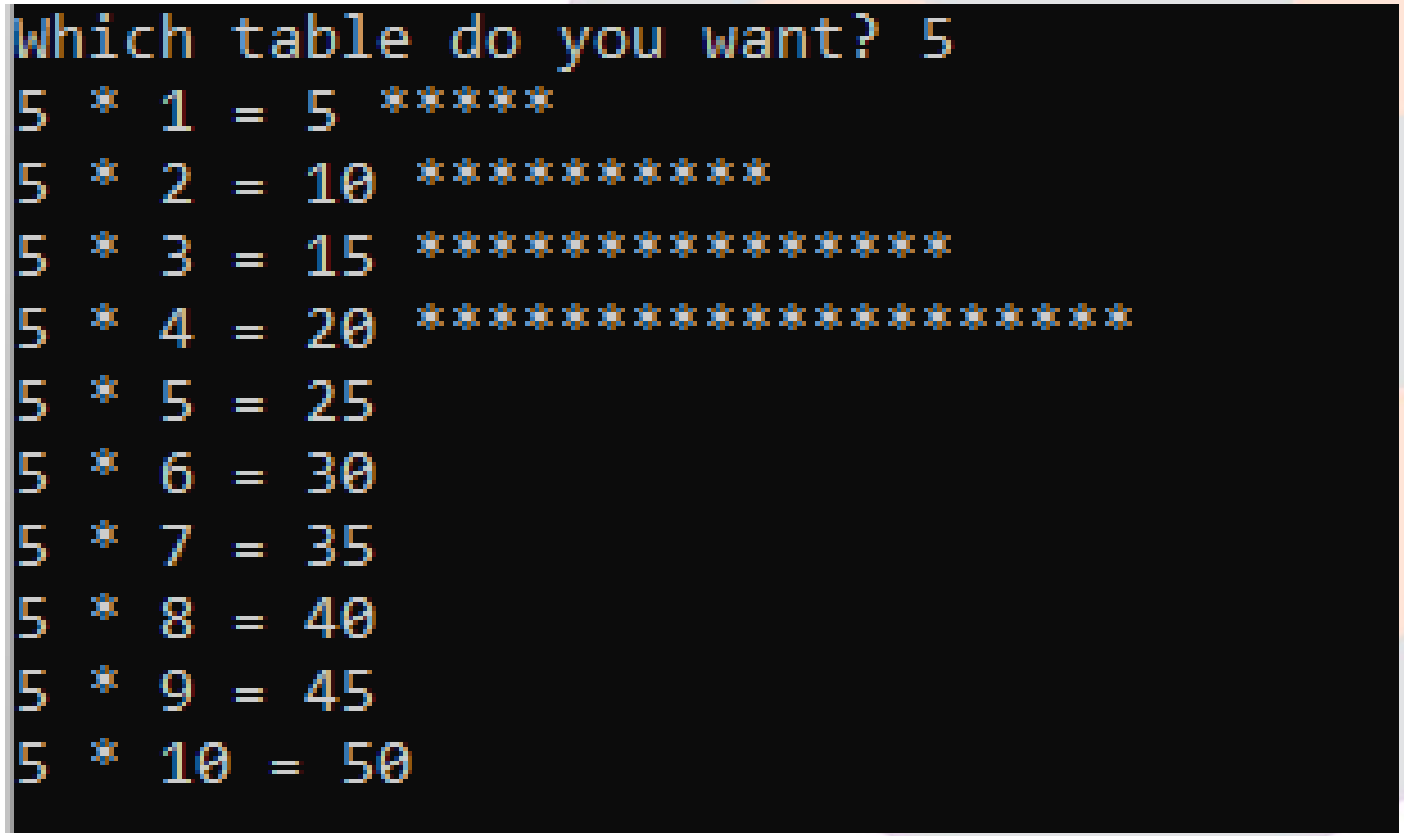


Print number of stars of the value of table

```c
#include<stdio.h>
int main(){
    int n;
    printf("Which table do you want? ");
    scanf("%d",&n);
    for(int i=1;i<=10;i++){
        printf("%d * %d = %d ",n,i,n*i);
        int stars = n*i;
        for(int k=0;k<stars;k++){
            printf("*");
        }
        printf("\n");
    }
    return 0;
}
```

We will discuss more programs Later

Don’t print stars if the stars count is more than 20

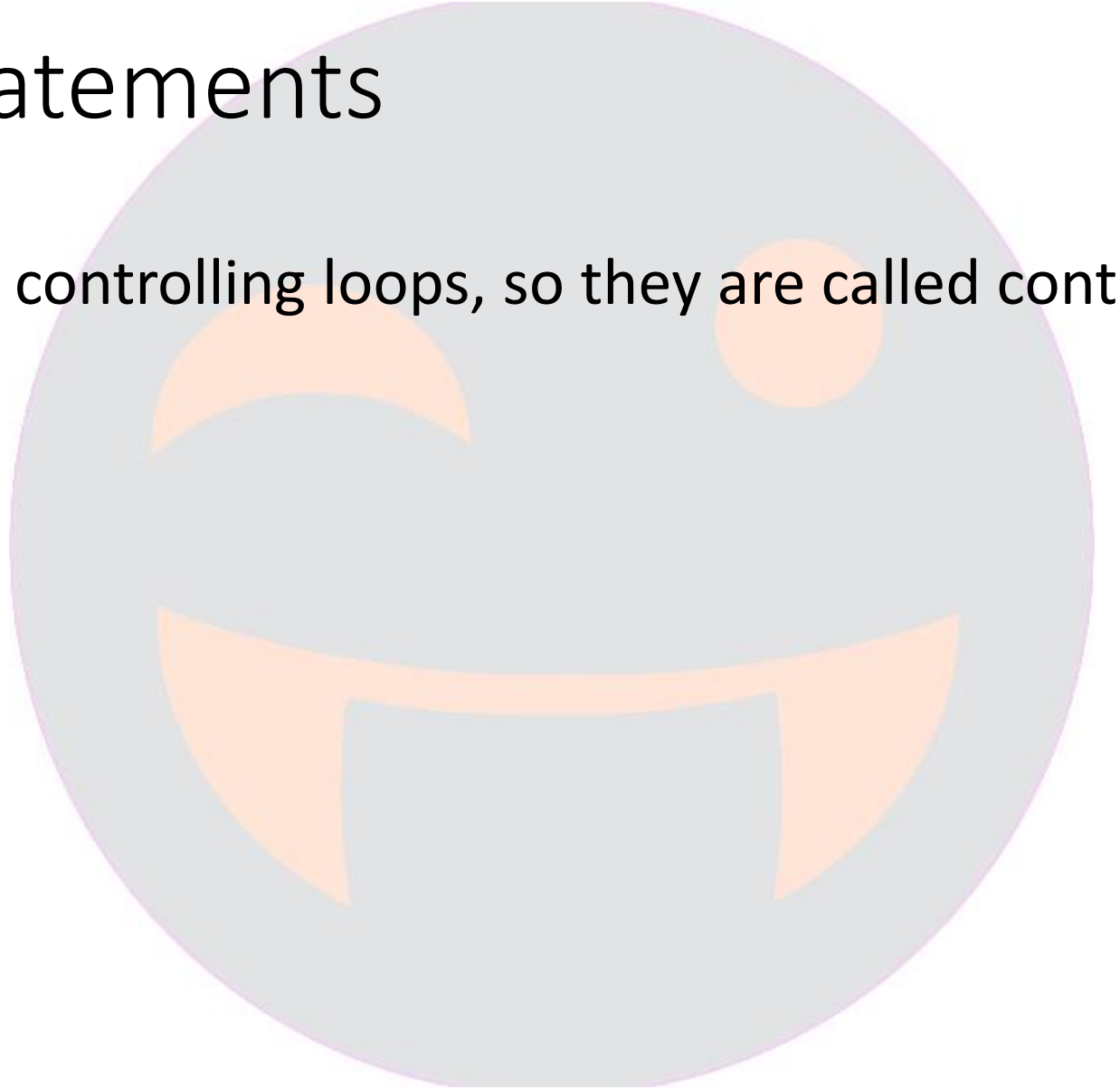


Don’t print stars if the stars count is more than 20

```c
#include<stdio.h>
int main(){
    int n;
    printf("Which table do you want? ");
    scanf("%d",&n);
    for(int i=1;i<=10;i++){
        printf("%d * %d = %d ",n,i,n*i);
        int stars = n*i;
        for(int k=0;k<stars;k++){
            if(stars>20)
                break;
            printf("*");
        }
        printf("\n");
    }
    return 0;
}
```

Control statements

- Since they are controlling loops, so they are called control statements
- break
- continue



break;

- Used to **exit the loop**
- break can be used to exit from any kind of loop(for, while, do while)

```
Which table do you want? 5
5 * 1 = 5 *****
5 * 2 = 10 **********
5 * 3 = 15 ***************
5 * 4 = 20 ********************
5 * 5 = 25
```

```c
#include<stdio.h>
int main(){
    int n;
    printf("Which table do you want? ");
    scanf("%d",&n);
    for(int i=1;i<=10;i++){
        printf("%d * %d = %d ",n,i,n*i);
        int stars = n*i;
        if(stars>20)
                break;
        for(int k=0;k<stars;k++){
            printf("*");
        }
        printf("\n");
    }
    return 0;
}
```

continue

- Used to send the control directly to the condition and to **continue the loop process**

- **Skips the current loop or statements** and enters into a new loop or condition.

Stop printing stars if the value is more than 20 but print the table

```
Which table do you want? 5
5 * 1 = 5 *****
5 * 2 = 10 **********
5 * 3 = 15 ***************
5 * 4 = 20 ********************
5 * 5 = 25
5 * 6 = 30
5 * 7 = 35
5 * 8 = 40
5 * 9 = 45
5 * 10 = 50
```

Stop printing stars if the value is more than 20 but print the table

```c
#include<stdio.h>
int main(){
    int n;
    printf("Which table do you want? ");
    scanf("%d",&n);
    for(int i=1;i<=10;i++){
        printf("%d * %d = %d ",n,i,n*i);
        int stars = n*i;
        if(stars>20){
            printf("\n");
            continue;
        }
        for(int k=0;k<stars;k++){
            printf("*");
        }
        printf("\n");
    }
    return 0;
}
```

Statements below continue are skipped

Test for you

```c
#include<stdio.h>
int main(){
    int n;
    printf("Enter a number");
    scanf("%d",&n);
    for(int i=1;i<=n;i++){
        if(i==5){
            break;
        }
        printf("%d\n",i);
    }
    return 0;
}
```

```
Enter a number10
1
2
3
4
```

Test for you

```c
#include<stdio.h>
int main(){
    int n;
    printf("Enter a number");
    scanf("%d",&n);
    for(int i=1;i<=n;i++){
        if(i==5){
            continue;
        }
        printf("%d\n",i);
    }
    return 0;
}
```

```
Enter a number10
1
2
3
4
6
7
8
9
10
```

What next?

- Let us see 15 programs using the topics we learned so far
- Till then **continue** liking our videos
- Don’t **break** the relationship with our channel 😍 😍 😍