NeuroCode Livestream

**TOPICS:**

* Switch Statement
* For Loop
* While Loop
* Do-While Loop
* Prime Numbers

Link of livestream:

<https://youtube.com/live/qLcDOQfI9Ss>

Link of playlist for all livestreams(in sequence):

<https://www.youtube.com/watch?v=lZ9Kqc451iM&list=PLixLvIo_nVYN5A44MN3cBz_9cWqP9vHzm>

Switch statement

A switch statement in C++ is a control structure used to make decisions. It allows a variable to be tested for equality against a list of values. Each value is called a "case", and the variable being switched on is checked for each case.

* Almost same as if else
* It checks for specific values only
* It does not check range
* It is useful when you want to run different block of code without writing if-statement again and again.

***Structure***

Switch(variable based on which we have to check multiple values)

{

case number or character :

//block of code

break;

default:

// it is just like *‘else’* part in *‘if-else’*

*}*

**PROGRAM 1:** write a program to print day based on number of a day. 1 for Monday … 7 for Sunday:

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int day;
6. cout<<"Enter number of day: ";
7. cin>>day;
8. switch(day)
9. {
10. case 1:
11. cout<<"Monday";
12. break;
13. case 2:
14. cout<<"Tuesday";
15. break;
16. case 3:
17. cout<<"Wednesday";
18. break;
19. case 4:
20. cout<<"Thursday";
21. break;
22. case 5:
23. cout<<"Friday";
24. break;
25. case 6:
26. cout<<"Saturday";
27. break;
28. case 7:
29. cout<<"Sunday";
30. break;
31. default:
32. cout<<"You entered invalid day!";
33. }
34. return 0;
35. }

DIFFERENCE BETWEEN SWITCH AND IF ELSE

|  |  |  |
| --- | --- | --- |
| ­ | *SWITCH* | *if-else* |
| Syntax & Structure | Tests a variable against a set of cases. | Executes blocks based on boolean conditions. |
| Readability | More readable with many specific values for a single variable. | Can be less readable with many conditions. |
| Type of Comparison | Limited to equality comparison against constants. | Supports a range of comparisons (equality, relational, etc.). |
| Default Behavior | Has a default case for unmatched values. | An else block serves for unmatched conditions. |
| Flexibility | Limited to integral and character types. | Can handle a wide range of data types and complex conditions. |
| Range Testing | Cannot test ranges of values. | Can evaluate and compare ranges. |
| Fall-through Behavior | Possible if break is omitted, can lead to bugs. | Not applicable. |
| Logical Conditions | Cannot evaluate logical conditions (&&,||) | Can evaluate logical conditions (&&,||) |

**PROGRAM 2:** Make a calculator program that take + - \* / % as input and then perform the calculation on given numbers!

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. //Make a calculator program that take + - \* / %
6. //as input and then perform the calculation
7. //on given numbers!
8. int num1,num2;
9. char inp; //we will store operator in it
10. cout<<"Enter num1: ";
11. cin>>num1;
12. cout<<"Enter num2: ";
13. cin>>num2;
14. cout<<"Enter + - \* / %: ";
15. cin>>inp;
16. switch(inp)
17. {
18. case '+':
19. cout<<"ans is : "<<num1+num2<<endl;
20. break;
21. case '-':
22. cout<<"Ans is : "<< num1 - num2 <<endl;
23. break;
24. case '\*':
25. cout<<"Ans is : "<< num1 \* num2 <<endl;
26. break;
27. case '/':
28. cout<<"ans is : "<< float(num1) / float(num2) <<endl;
29. break;
30. case '%':
31. cout<< "ans is : "<<num1 % num2 <<endl;
32. break;
33. default:
34. cout<<"You entered invalid operator! "<<endl;
35. }
36. return 0;
37. }

LOOPS IN C++

Used to repeat a block of code until a certain condition is met.

3 types:

1. For loop
2. While loop
3. Do-while loop

FOR LOOP

We use for loop when we know

how many times we have to

repeat a block of code.

Syntax:

for (initialization; condition; increment/decrement) {

// code block to be executed

}

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. //print "university is off on monday" 10 times;
6. //for(initialization;condition ; increment)
7. // i++ is same as i = i+1
8. // 1 2 4
9. for(int i = 0 ; i < 0 ; i++)
10. {
11. // 3
12. cout<<"University is off on monday "<< i<<endl;
13. }
14. return 0;
15. }

WHILE LOOP

We use while loop when we want

to repeat a block of code until

a certain condition is met.

Syntax:

while (condition) {

// code block to be executed

}

In while, we write condition on the top. So, compiler checks the condition first and then it executes the code of condition is true

**PROGRAM 3:** ask user to enter number on repeat until he enter positive number

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. // we have to keep taking input from user
6. // until the user enter a positive number
7. int num;
8. cout<<"Enter a positive number: ";
9. cin>>num;
10. while (num < 0)
11. {
12. cout<<"Error! Enter a positive number: ";
13. cin>>num;
14. }
15. cout<<"You entered a positive number : "<<num<<endl;
16. return 0;
17. }

DO-WHILE LOOP

Same as while loop.

The only difference that we

write condition at the end.

Syntax:

do {

// code block to be executed

} while (condition);

It means that when compiler first comes to do-while loop, it will run block of code once, and then at the end of block it will check the condition.

**PROGRAM 4:** ask user repeatedly to enter a number until user enter a positive number!

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. // we have to keep taking input from user
6. // until the user enter a positive number
7. int num;
8. do
9. {// this code will run at least once
10. cout<<"Enter a positive number: ";
11. cin>>num;
12. }while (num < 0);
13. cout<<"You entered a positive number : "<<num<<endl;
14. return 0;
15. }

**PROGRAM 5:** Write a program in C++ to find the first 10 natural numbers.   
Sample output:  
The natural numbers are:  
1 2 3 4 5 6 7 8 9 10

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. for(int i=1 ; i<=10 ; i++)
6. {
7. cout<<"Natural number: "<<i<<endl;
8. }
9. return 0;
10. }

**Program 6:** Write a program in C++ to find the sum of the first 10 natural numbers.

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int sum = 0;
6. for(int i=1 ; i<=10 ; i++)
7. {
8. cout<<"Natural number: "<<i<<endl;
9. sum = sum + i;
10. cout<<"value of sum varibale : "<<sum<<endl;
11. }
12. cout<<"Sum of first 10 natural numbers is : " <<sum<<endl;
13. return 0;
14. }

**PROGRAM 7:** check if a number is divisible by 5

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int num;
6. //how to check if a number is divisible by 5
7. cout<<"Enter a number to check if it is divisible by 5: ";
8. cin>>num;
9. if(num%5 == 0)
10. {
11. cout<<"divisible";
12. }
13. else
14. {
15. cout<<"not divisible";
16. }
17. return 0;
18. }

**PROGRAM 8:** CHECK IF A NUMBER IS PRIME NUMBER

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. // a prime number is only
6. // divisible by 1 and
7. // the number itself
8. // it has only 2 number of divisibles
9. int num = 45;
10. //step 1: we have to run loop from 1 till number
11. int numOfDiv = 0;
12. for(int i = 1; i<=num ; i++)
13. {
14. // step 2: we will count number of divisibles
15. if(num % i == 0)
16. {
17. numOfDiv = numOfDiv + 1;
18. }
19. }
20. // step 3: check if there are only 2 number of divisibles
21. if(numOfDiv == 2)
22. {
23. cout<< num << " is a Prime number! "<<endl;
24. }
25. else
26. {
27. cout<< num << " is not a prime number! "<<endl;
28. }
29. return 0;
30. }

PROGRAM 9: Ask user on repeat to check a prime number and terminate(end) program when user enter 0

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. //keep asking user to check prime number until user enter 0
6. int num=1;
7. int numOfDiv = 0;
8. while( num != 0)
9. {
10. cout<<"Enter a number to check prime(0 to terminate): ";
11. cin>>num;
12. numOfDiv = 0;
13. for(int i = 1; i<=num ; i++)
14. {
15. if(num % i == 0)
16. {
17. numOfDiv = numOfDiv + 1;
18. }
19. }
20. if(numOfDiv == 2)
21. {
22. cout<< num << " is a Prime number! "<<endl;
23. }
24. else
25. {
26. cout<< num << " is not a prime number! "<<endl;
27. }
28. }
29. return 0;
30. }