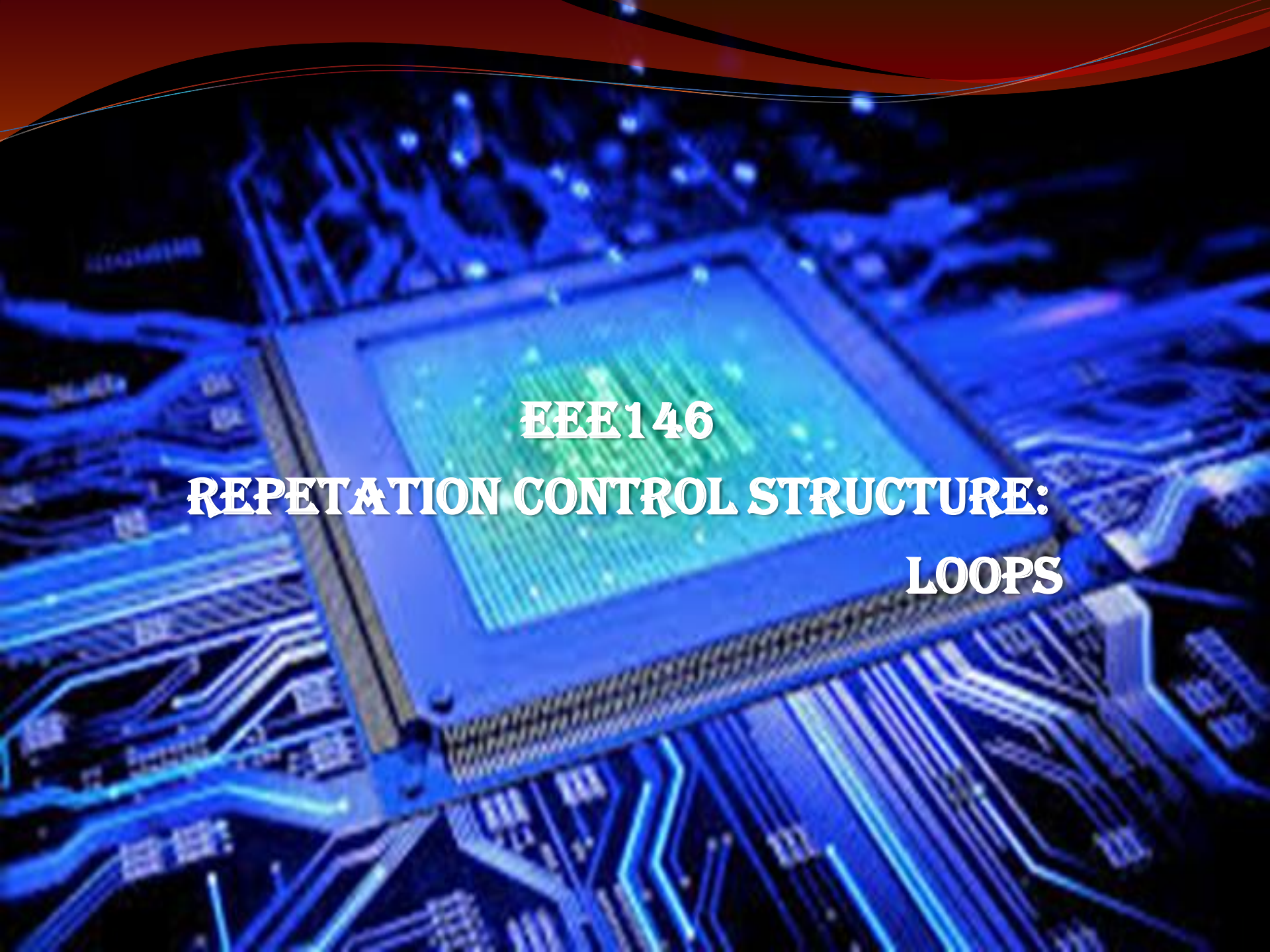
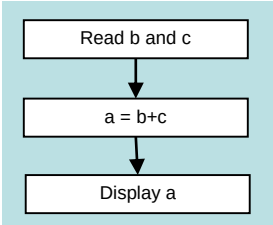
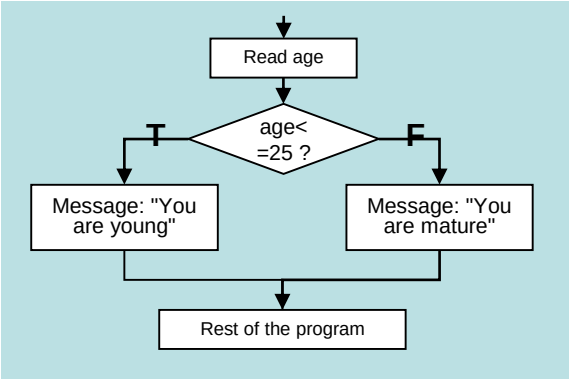

EEE146
REPETATION CONTROL STRUCTURE:
LOOPS

Control of flow

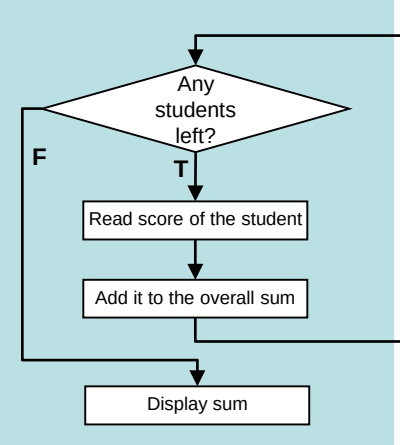
- We learned that default flow of instructions is sequential.



- Then, we learned how to control the flow using "if" and "switch."



- Now, we will learn how to *repeat* a group of instructions.



Types of loops

- There are three types of loops:
  - "for" loop
  - "while" loop
  - "do-while" loop
- You can implement anyone of these loops using the others (and possibly an additional if statement and duplication of some statements).
  - The idea is to use the type of loop that is best suited for your problem.

General structure of a loop

- A loop is typically composed of three parts:
  - the statement block which is to be repeated;
  - a control mechanism to decide whether the statement block should be repeated once more (based on an expression);
  - an update mechanism that affects the value of the control expression (to avoid infinite loops).
- The update mechanism may be embedded in the statement block.

General structure of a loop

- One execution of the statement block is called an iteration.
- Thus, a loop *iterates* the statement block several times.
- Make sure:
  - it is possible to enter the loop;
  - it is possible to get out of the loop, once you enter it.

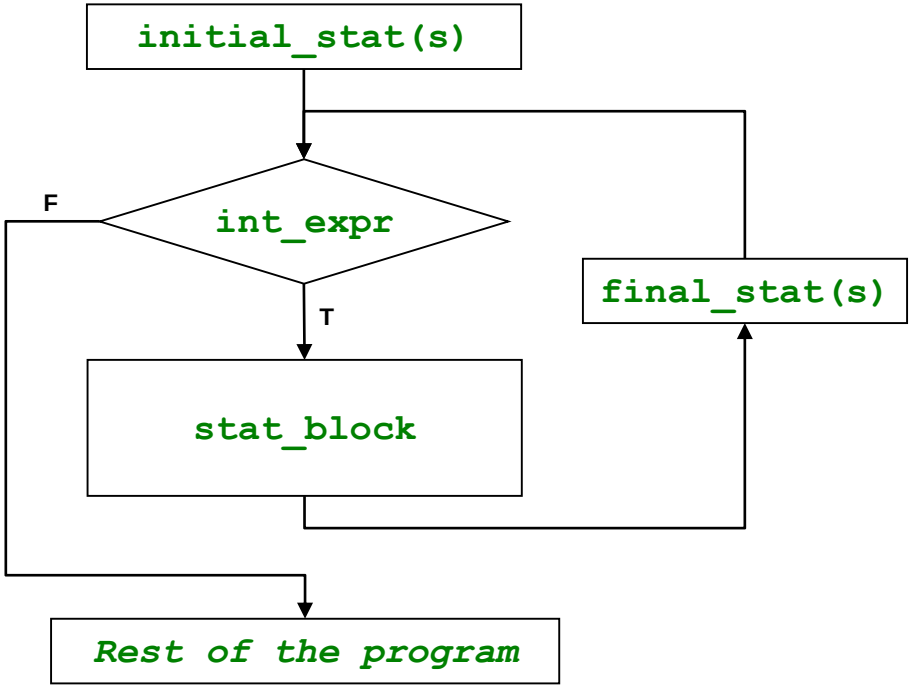
For loop

- Use for loop if you know the number of iterations.
  - You don't have to know the exact number of iterations; it is enough if you can express it.

For loop

- Syntax:

```
for (initial_stat(s);int_expr; final_stat(s))
    stat_block
```



For loop

- Note that initial and final statements as well as the integer expression are optional according to the syntax.
  - If **`initial_stat(s)`** is missing, start directly with the comparison.
  - If **`final_stat(s)`** is missing, go directly to the comparison after the execution of the statement block.
  - If **`int_expr`** is missing, the comparison is always true.
    - Make use of the **`break`** statement (to be discussed later).

for: Example 1

Write a code segment that prints the text "I didn't do my homework" 100 times with every line enumerated.

```
cout<<"1.  I didn't do my homework.\n";
cout<< "2.  I didn't do my homework.\n";
cout<< "3.  I didn't do my homework.\n";
cout<< "4.  I didn't do my homework.\n";
cout<< "5.  I didn't do my homework.\n";
cout<< "6.  I didn't do my homework.\n";
cout<< "7.  I didn't do my homework.\n";
cout<< "8.  I didn't do my homework.\n";
cout<< "9.  I didn't do my homework.\n";
cout<< "10. I didn't do my homework.\n";
cout<< "11. I didn't do my homework.\n";
cout<< "12. I didn't do my homework.\n";
cout<< "13. I didn't do my homework.\n";
cout<< "14. I didn't do my homework.\n";
cout<< "15. I didn't do my homework.\n";
.
.
cout<< "91. I didn't do my homework.\n";
cout<< "92. I didn't do my homework.\n";
cout<< "93. I didn't do my homework.\n";
cout<< "94. I didn't do my homework.\n";
cout<< "95. I didn't do my homework.\n";
cout<< "96. I didn't do my homework.\n";
cout<< "97. I didn't do my homework.\n";
cout<< "98. I didn't do my homework.\n";
cout<< "99. I didn't do my homework.\n";
cout<< "100.I didn't do my homework.\n";
```

for: Example 1 *(cont'd)*

Lazy student's solution ☺

```
int i;

for (i=1; i<=100; i++)
  cout<< i << ".I didn't do my homework.\n";
```

for: Example 2

Find a^b. (a and b integers)

```
int a, b, result=1, i;


cin>> a >> b;
for (i=0; i<b; i++)
    result *= a;
```

for: Example 2 *(cont'd)*

Same question, solved with fewer variable (but the value of `b` changes).

```
int a, b, result=1;


cin>> a >> b;
for (; b; b--)
    result *= a;
cout<<"result="<< result;
```

for: Example 3

Find the average of the midterm scores of the students in EEE145.

```
int i, sum, no_stu, score;
float avg;
cin>> no_stu;
for (sum=i=0; i<no_stu; i++)
{
    cin >> score;
    sum += score;
}
avg = sum/no_stu;
```

(float)

What is wrong here?

What else?

What if "no_stu" is zero?

for: Example 4

Calculate BMI (Body Mass Index) of all students in class ☹

- weight/height

```
for (i=0; i<100; i++)
{
    cin>> weight >> height;
    cout<<"BMI:"<< (float)weight/height;
}
```

What if "height" is zero?

for: Example 5

What are the differences between these code segments?

`for (i=0; i<5; i++)`
`  cout<< i<< ";";`
`cout<<"{"<< i <<"}";`
	`for (i=0; i<=5; i++)`
	`  cout<< i<< ";";`
	`cout<<"{"<< i <<"}";`
		`for (i=1; i<5; i++)`
		`  cout<< i<< ";";`
		`cout<<"{"<< i <<"}";`

`0;1;2;3;4;{5}`	`0;1;2;3;4;5;{6}`	`1;2;3;4;{5}`

`for (i=1; i<=5; ++i)`
`  cout<< i<< ";";`
`cout<<"{"<< i <<"}";`
	`for (■; i<5; i++)`
	`  cout<< i<< ";";`
	`cout<<"{"<< i <<"}";`
		`for (i=0;■; i++)`
		`  cout<< i<< ";";`
		`cout<<"{"<< i <<"}";`

`1;2;3;4;5;{6}`	`Starts from anything ...;3;4;{5}` `OR MAYBE SOMETHING LIKE {795}`	`0;1;2;...∞`

`for (i=0; i<5;■)`
`  cout<< i<< ";";`
`cout<<"{"<< i <<"}";`
	`for (i=0; i<5;■)`
	`  cout<< i++<< ";";`
	`cout<<"{"<< i <<"}";`
		`for (i=0; i++<5;■)`
		`  cout<< i<< ";";`
		`cout<<"{"<< i <<"}";`

`0;0;0;0;...∞`	`0;1;2;3;4;{5}`	`1;2;3;4;5;{6}`

for: Example 6

Now, compare these code segments.

`for (i=7; i<5; i++)`
`  cout<< i<< ";";`
`cout<<"{"<< i <<"}";`
	`for (i=7;++i<5;■)`
	`  cout<< i<< ";";`
	`cout<<"{"<< i <<"}";`
		`for (i=7;i++<5;■)`
		`  cout<< i<< ";";`
		`cout<<"{"<< i <<"}";`

`{7}`	`{8}`	`{8}`

`for (i=7; ++i<5; ++i)`
`  cout<< i<< ";";`
`cout<<"{"<< i <<"}";`
	`for (i=0; i<5; ++i)`
	`  cout<< ++i<< ";";`
	`cout<<"{"<< i <<"}";`

`{8}`	`1;3;5;{6}`

for: Example 7

- What if Carl Friedrich Gauss knew how to program when he was in elementary school?
  - Let's be more generic; add numbers from 1 to n.

```
cin >> n;
for (sum=0, i=1; i<=n; i++)
    sum += i;
```

- Of course Gauss would be clever enough to do

```
cin >> n;
sum = n * ((n+1)/2);
```

for: Example 8

Find whether a number is perfect or not.

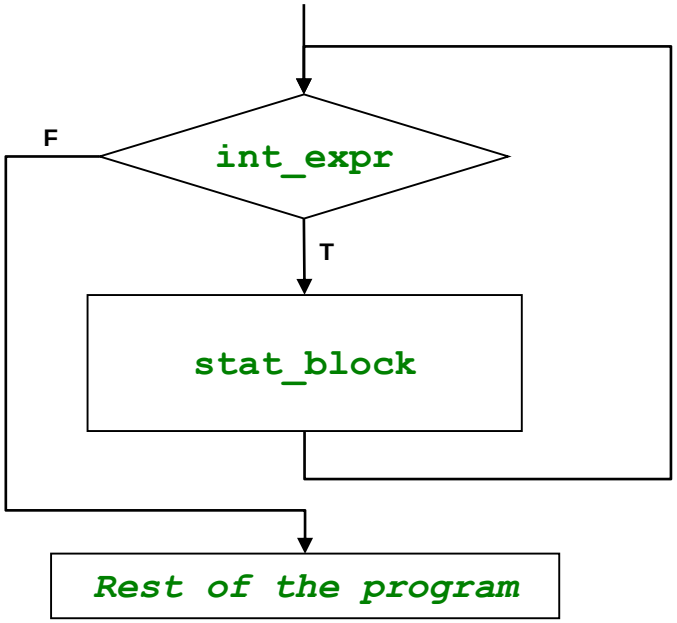
(A number is perfect if sum of its positive divisors except itself is equal to itself. Eg: 6=1+2+3; 28=1+2+4+7+14)

```
int number, i, sum = 0;
cin >> number;
for (i = 1; i <= number / 2 ; i++)
    if (number % i == 0)
        sum += i;
if (number == sum)
    cout<< number << "is a perfect number";
else
    cout<< number << "is not a perfect number";
```

While loop

- Use while loop if the statement block should be executed as long as a condition holds.
- Syntax:

```
while (int_expr)
    stat_block
```



while: Example 1

Find the average of a sequence of integers terminated with a negative value.

```
int sum=0, n, count=0;
float avg;

cin >> n;
while (n>=0)
{
    sum += n;
    count++;
    cin >> n;

}
avg = (count)?(float)sum/count:0;
```

while: Example 2

Consider a type of cell that reproduces by mitosis. Assume each cell divides into two cells every second. Display the number of cells after each second for 100 seconds. Start with one cell.

```
int n=1, t=0;


while (t<=100)
{
    cout<<"t= " << t <<"n= "<<n<<endl;
    n *= 2;
    t++;
}
```

while: Example 3

Assume the user enters a sequence of 1s and 0s. Any other character marks the end of input. Take 1's complement of the input.

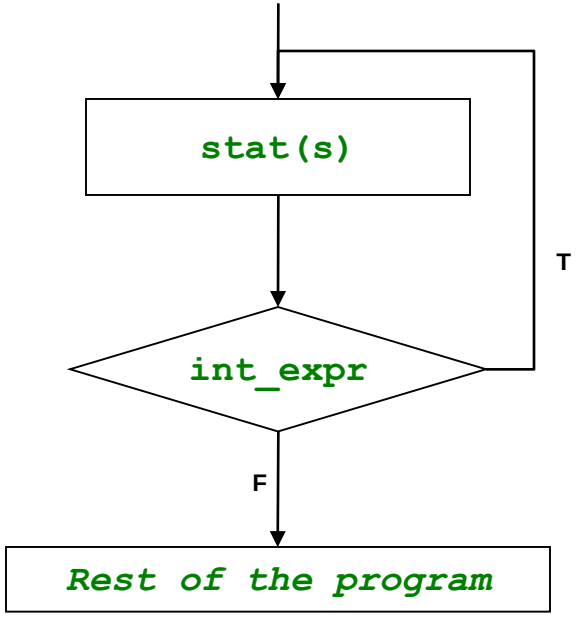
```
char ch;

ch=getchar();
while ((ch=='0') || (ch=='1'))
{
    cout<< !(ch-'0');
    ch=getchar();
}
```

Do-while loop

- Similar to while loop.
  - Only difference: Condition is checked after the execution of every iteration.
- Syntax:

```
do
{
    stat(s)
}
while (int_expr);
```



do-while: Example 1

Solve the previous example, this time using do-while.

Find the average of a sequence of integers terminated with a negative value.

```
int sum=0, n, count=0;
float avg;
do
{
    cin>> n;
    if (n>=0)
    {
        sum += n;
        count++;
    }
} while (n>=0);
avg = (count)?(float)sum/count:0;
  cout<< avg <<endl;
```

Note that we repeated the condition in the if statement, so it was not a good idea to solve this problem using do-while.

break statement

- It is possible to terminate a loop prematurely.
- Remember the **`break`** statement we discussed before.
  - **`break`** breaks the innermost loop or switch statement.

break: Example

Read 100 integers and find their product. However, if one of the numbers is non-positive, stop input.

```
long int mul=1;
int i, num;

for (i=0; i<100; i++)
{
    cin>>num;
    if (num<=0)
        break;
    mul *= num;
}
cout<<mul<<endl;
```

continue statement

- It is possible to skip the rest of an iteration and continue with the next iteration (if any).
  - In the for loop, **`continue`** jumps to the final statement.
  - In the while and do-while loops, **`continue`** jumps to the condition expression.

continue: Example 1

Consider the following two code segments.

Assume input is: `5 -4 3 2 -5 8 9 1`

```
sum = i = 0;
while (i < 6)
{
    cin>>no;
    if (no <= 0)
        continue;
    sum += no;
    i++;
}
/* sum becomes 28 */
```

```
sum = 0;
for (i = 0; i < 6; i++)
{
    cin>>no;
    if (no <= 0)
        continue;
    sum += no;
}

/* sum becomes 18 */
```

continue: Example 2

Find the number of passengers/car for every flat in a building with 40 flats.

```
for (i=0; i<40; i++)
{
   cin>>no_cars;
    if (no_cars==0)
       continue;
    cin >> no_residents;
   cout<< (float)no_residents/no_cars
}
```

Example: Nested loops

You can nest the loops as you do with if statements.

For each iteration of the outer loop, the inner loop is executed from beginning to the end.

```
int count=0,i,j;
for(i=1;i<=5;i++)
    for(j=0;j<=4;j++)
        count++;
```

```
Number of iterations of outer loop:5
Number of iterations of inner loop:5
count is incremented 5×5 times

count = 25
```

Nested loops: Example 1

Draw a right triangle using '*' character. Number of lines is read as input.

```
#include <iostream>

int main()
{
   int line, i, j;

   cout<<"Enter the height :";
   cin >> line;
   for (i = 1; i <= line; i++)
   {
      for (j = 1; j <= i ; j++)
         cout<<"*";
      cout<<"\n";
   }
   return 0;
}
```

Nested loops: Example 2

Draw a right triangle using '*' character. Number of lines is read as input.

```
#include <iostream>

int main()
{
   int line, i, j;

   cout<<"Enter the height :";
   cin >> line;
   for (i = 1; i <= line; i++)
   {
      for (j = 0; j <= line-i ; j++)
         cout<<"*";
      cout<<"\n";
   }
   return 0;
}
```

Nested loops: Example 3

Draw an isosceles triangle using '*' character. Number of lines is read as input.

```
#include <iostream>

int main()
{
   int line, i, j;

   cout<<"Enter the height :";
   cin >> line;
   for (i = 1; i <= line; i++)
   {
      for (j = 0; j < line - i; j++)
         cout<<" ";
      for (j = 0; j < i * 2 - 1; j++)
         cout<<"*";
      cout<<"\n";
   }
   return 0;
}
```

Example

- Read an integer and print its digits in reverse order.

```
#include <iostream>

int main()
{  int num, digit;

   cin>>num;
   while (num)
   {  digit=num%10;
      num /= 10;
      cout<<digit;
   }
   return 0;
}
```

Example

- Solve the question by considering only the first four digits after the decimal point to be significant. Eg: If the input is 35.794678, the output should be 7946.35.

```
#include <stdio.h>

int main()
{   float num, dec, whole;
    int i;

    cin>>num;
    whole = (int) num;
    dec = num-whole;
    while (whole>1)
       whole /= 10;
    for (i=0; i<4; i++)
      dec *= 10;
    num = (int)dec+whole;
    cout<<num<<endl;
}
```

Example

- Write a C++ program that will calculate

$$\log(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \ldots \qquad |x| < 1$$

using series representation. Note that this series converges if

```
#include <iostream>
#include <cmath>
using namespace std;

int main()
{
  double sum, term, x=2.0;
  int n;
  while(x>1.0 || x<-1.0){
    cout<<"Enter a value between -1 and+1:";
    cin>>x;
  }
  term=sum=x; n=1;
  while(fabs(term) > 1.0e-8){
    ++n;
    term= -term*(x/n)*(n-1);
    sum += term;
  }
  cout<<"The sum of the series:"<< sum<<endl;
  cout<<"Number of terms in the series:"<< n<<endl;
  cout<<"cmat function result:"<<log(1+x)<<endl;
  return 0;
}
```

Homework

- The exponential function can be represented by infinite series

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots + \frac{x^n}{n!}$$

Write a C++ program to calculate exp(x) using the given series.

Homework

- The trigonometric **sine** function can be represented by infinite series

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots$$

Write a C++ program to calculate sin(x) using the given series.