

មេរៀនទី១

C programming

គោលដៅមេរៀន

- របៀបបង្កើត កម្មវិធី C
- ធ្វើការរៀបចំកម្មវិធី C
- សរសេរកម្មវិធីសម្រាប់បង្ហាញនៅលើscreen

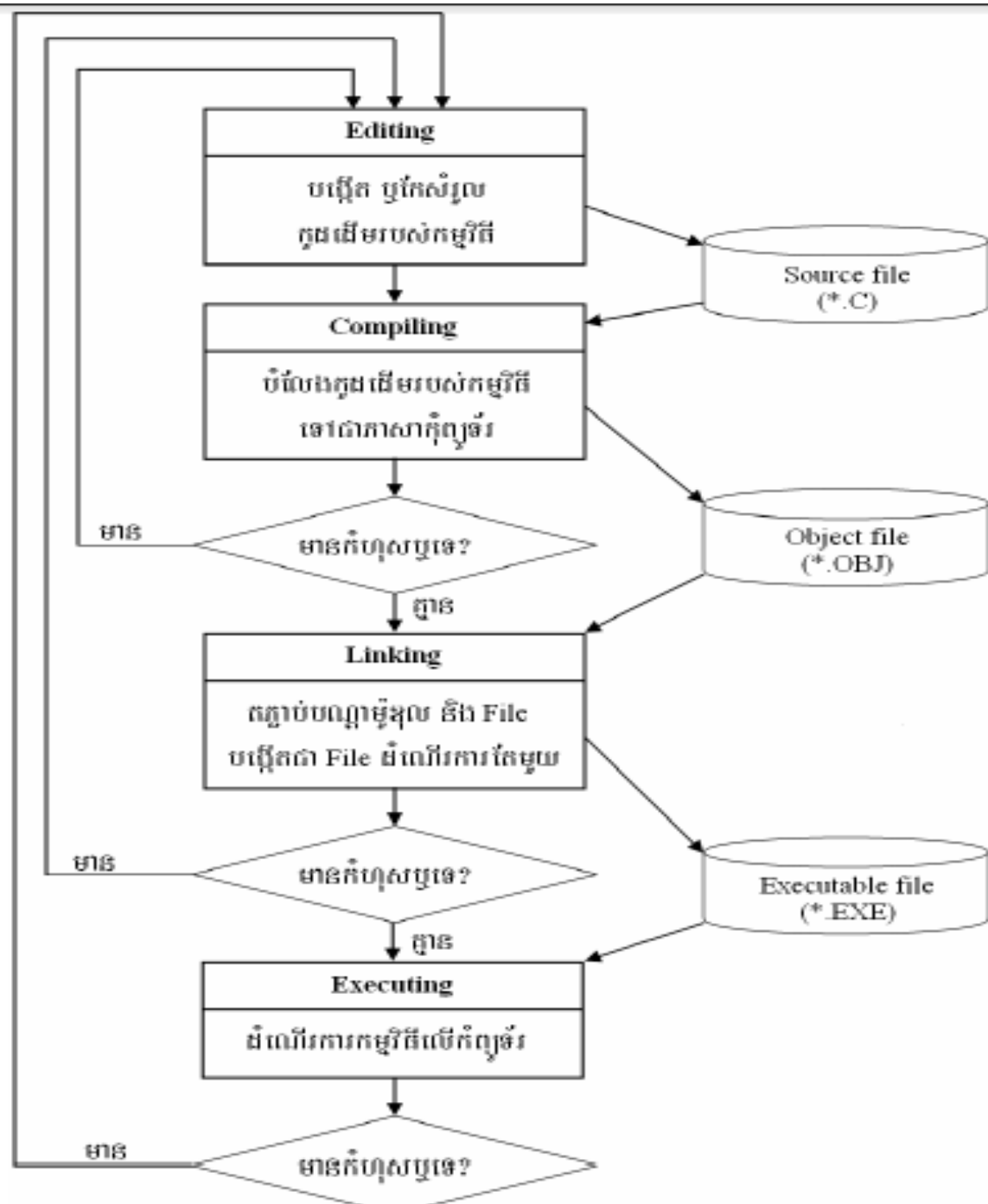
បង្កើតកម្មវិធីដោយភាសា C

- ដើម្បីបង្កើតកម្មវិធីមួយដោយភាសា C ត្រូវឆ្លងកាត់ 4 ដំណាក់កាល ៖
- Editing
- Compiling
- Linking
- Executing

Editing

- ជាដំណាក់កាលដំបូងបំផុត គឺសរសេរកូដរបស់កម្មវិធី (source code) និងដាក់ឈ្មោះអោយកម្មវិធី
save file .c
- Compiling
- ជាកម្មវិធីមួយដែលមានមុខងារបកប្រែ និងបំប្លែងកូដដើមនៃកម្មវិធីអោយទៅជាភាសាមួយដែលកុំព្យូទ័រ អាចយល់បាន។ (បំប្លែងFile.obj)

- Linking
- ជាអង្គតភ្ជាប់ បណ្តាFile ដែលបានបំប្លែងដោយ
Compiler និង library file .
- Execution
- ជាដំណាក់កាលដែលកម្មវិធីដំណើរការលើកុំព្យូទ័រ
.exe



បណ្តាញត្រួតពិនិត្យនៅក្នុងភាសា C

- ការប្រើប្រាស់តួអក្សរនៅក្នុងភាសា C
- គ្រប់ភាសាសរសេរកម្មវិធីទាំងអស់ សុទ្ធតែកើតឡើងដោយសារសំណុំនៃអក្សរ ដែលតួអក្សរនីមួយៗត្រូវបានផ្គុំចូលគ្នាបានជាពាក្យ និង statement ។

- 26 អក្សរធំ (upper case) : A , B ,CZ
- 26 អក្សរតូច (Lower case) :a , b cz
- 10 តួលេខ (Number) : 0,1 ,2,3,4,5,6,7,8,9
- សញ្ញាពិសេស : ,.:/ ?{[]}%^& * ()<>'""
- សញ្ញា Underscore : _
- ❖ ចំណាំ ក្នុងភាសា C អក្សរតូច និង អក្សរធំ មានន័យខុសគ្នា

បង្កើតកម្មវិធីដំបូង

```
#include<stdio.h>
```

Header file

```
int main( )
```

main function

```
{
```

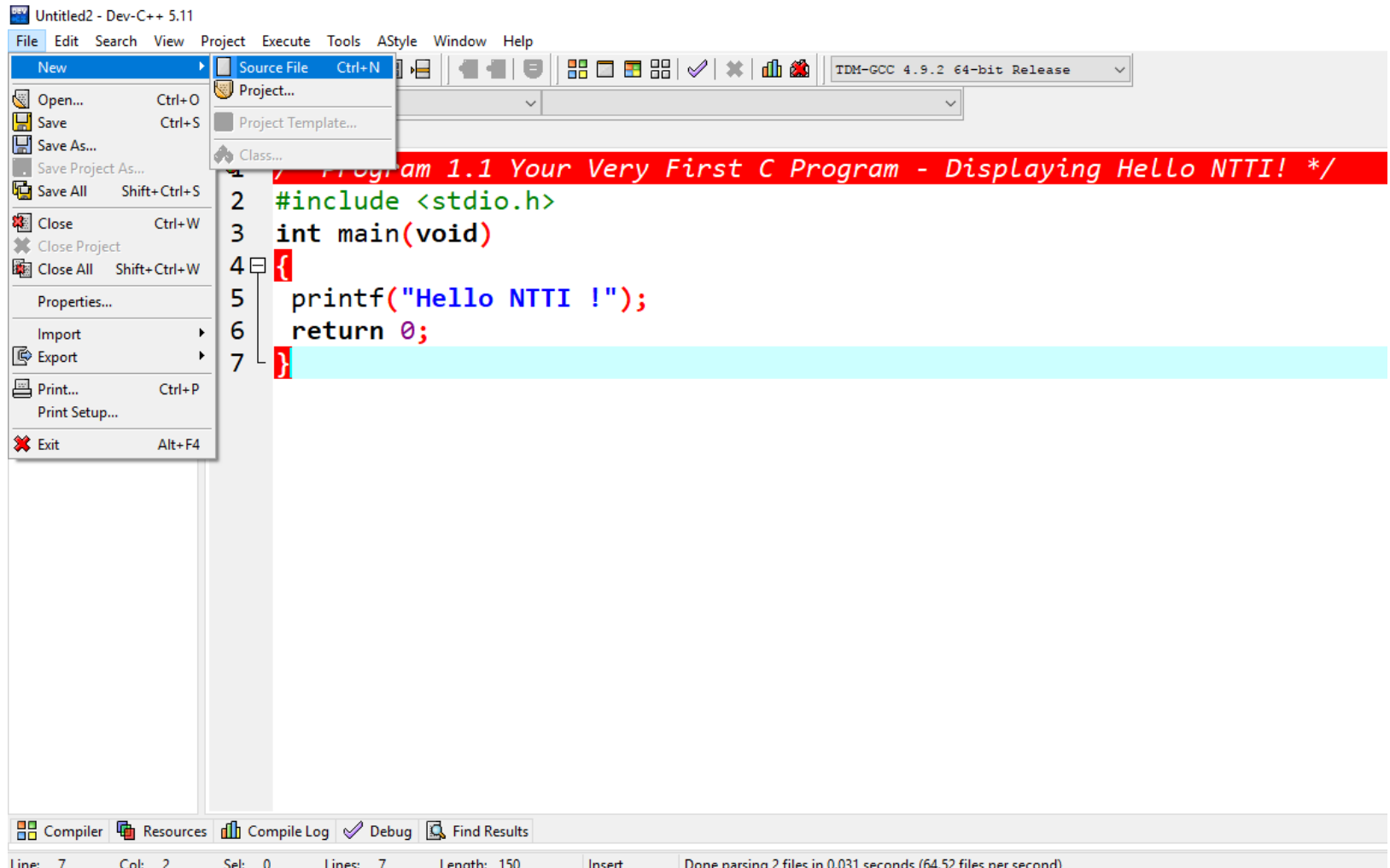
```
printf( "Welcom to C Language" );
```

```
return ( 0 );
```

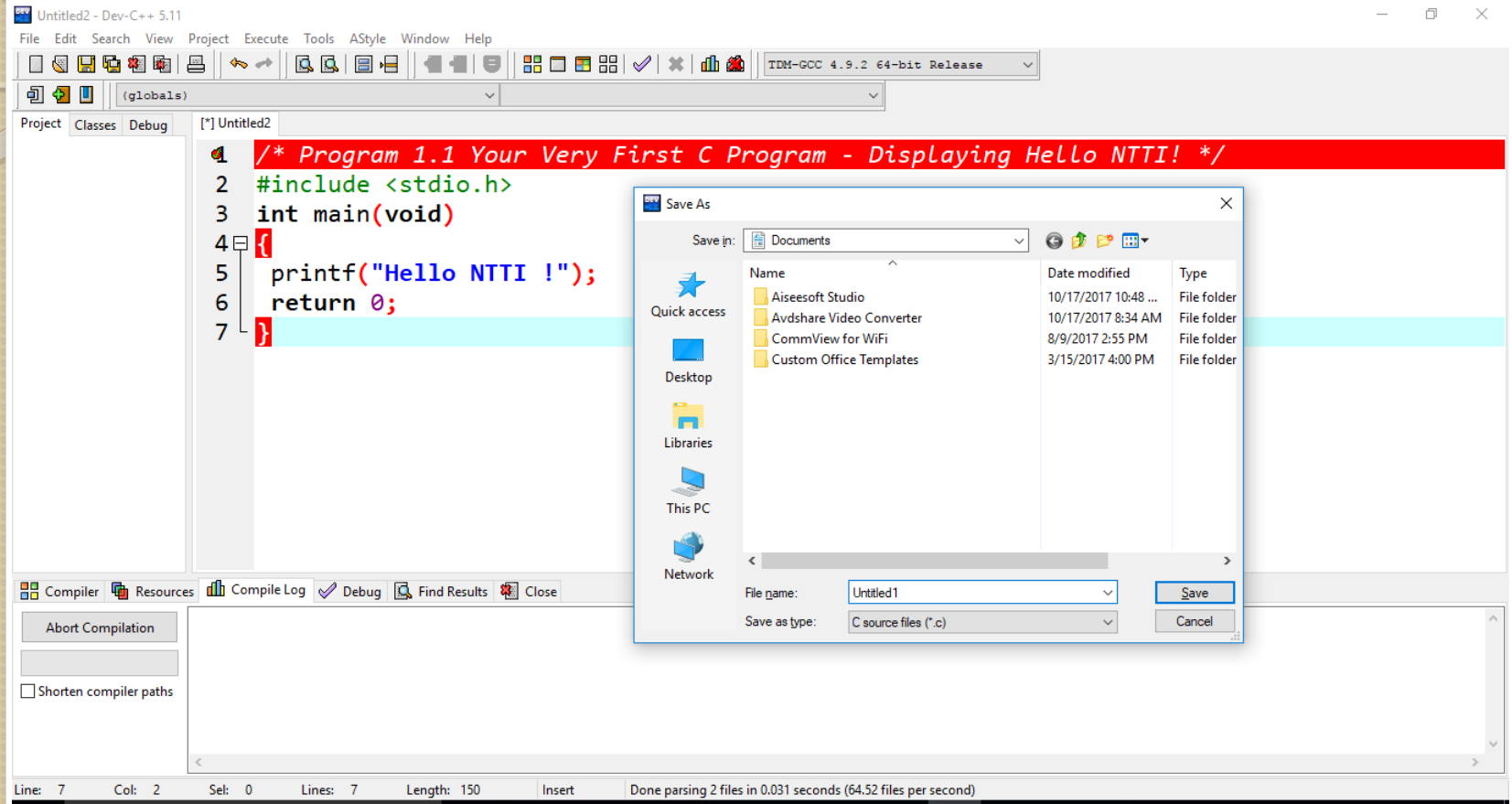
```
}
```

- # ជាអង្គចង្អុលបង្ហាញ compiler
- include សម្រាប់នាំcompiler អានអត្ថន័យ File ជាកំភ្លង < >
- stdio.h ជាheader file (standard input / output) ដែលផ្ទុកកូដរបស់អនុគមន៍ជាច្រើន printf , scanf , get ...
- main ()
- { } ជាអនុគមន៍ main ដែលត្រូវតែមានជាដាច់ខាតគ្រប់កម្មវិធីដោយសារត្រូវបានប្រើមុនគេ។
- Statement ជាបណ្តុំពាក្យត្រូវបានដំរៀបទៅតាមលំនាំណាមួយ និងមានមុខងារសម្រាប់អនុវត្តការងារនោះ។

បង្កើត File ផ្ទុកកូដដើមរបស់កម្មវិធី



Compile (F9)



មេរៀនទី២

- ប្រភេទទិន្នន័យ និង ប្រមាណវិធី
- Data type and Operator

គោលដៅមេរៀន

- បង្ហាញពីការប្រើប្រាស់អថេរនៅក្នុង Memory
- ពន្យល់ពីប្រភេទខុសគ្នារបស់អថេរដែលមានក្នុងភាសា C

- `/* Program 2.1 What is a Variable ? */`
- `#include <stdio.h>`
- `int main(void)`
- `{`
- `printf("My salary is $10000");`
- `return 0;`
- `}`

I. Data type

- Data Type: ជាប្រភេទទិន្នន័យដែលត្រូវបានកំណត់ឲ្យអថេរដើម្បីប្រើប្រាស់នៅលើ Memory។ ដូច្នេះនៅពេលដែលយើងចង់ប្រកាសអថេរដើម្បីប្រើប្រាស់ផ្ទុកទិន្នន័យនៅក្នុង C Programming Language យើងត្រូវគិតដល់ប្រភេទទិន្នន័យរបស់វា ដូចជា លេខ Number តួអក្សរ Character ឃ្លាប្រយោគ String។
- ប្រភេទទិន្នន័យនៅក្នុង C Language ដែលគេនិយមប្រើរួមមាន:

I. ប្រភេទចំនួនគត់

ប្រភេទទិន្នន័យ	ពាក្យគន្លឹះ	ចំនួន Byte	ដែនតំលៃ
Character	char	1	-128 ដល់ 127
Integer	int	2	-32768 ដល់ 32767
Short integer	short	2	-32768 ដល់ 32767
Long integer	long	4	-2147483648 ដល់ 2147483647
Unsigned character	unsigned char	1	0 ដល់ 255
Unsigned intger	unsigned int	2	0 ដល់ 65535
Unsigned short intger	unsigned short	2	0 ដល់ 65535
Unsigned long intger	unsigned long	4	0 ដល់ 4294967295

I. Data type (Continue)

- char: សំរាប់ផ្ទុកនូវតួអក្សរ និង ចំនួនលេខ(មិនអាចគណនាបាន) មានទំហំ 1byte ដែលមានតំលៃចាប់ពី -128 ទៅ 127។
- unsigned char: សំរាប់ផ្ទុកនូវតួអក្សរ ឬ លេខ(មិនអាចគណនាបាន) មានទំហំ 1byte ដែលមានតំលៃចាប់ពី 0 ទៅ 255។
- int (short int): សំរាប់ផ្ទុកនូវចំនួនគត់មានទំហំ 2byte មានតំលៃចាប់ពី -32768 ដល់ 32767។
- unsigned int: សំរាប់ផ្ទុកនូវចំនួនគត់មានទំហំ 2byte មានតម្លៃពី 0 ដល់ 65535។
- long: សំរាប់ផ្ទុកនូវចំនួនគត់មានទំហំ 4byte មានតំលៃពី -2147483648 ទៅដល់ 214783647។
- unsigned long: សំរាប់ផ្ទុកនូវចំនួនគត់មានទំហំ 4byte មានតំលៃពី 0 ដល់ 4254967294។

ចំនួនពិត

ប្រភេទចំនួន	ពាក្យគន្លឹះ	ចំនួន Byte	ដែនតំលៃ
Single-precision floating point	float	4	3.4E-38 ដល់ 3.4E+38
Double-precision floating point	double	8	1.7E-308 ដល់ 1.7E+308
Long double-precision floating point	long double	10	3.4E-4932 ដល់ 1.1E+4932

I. Data type (Continue)

- float: សំរាប់ផ្ទុកនូវចំនួនទសភាគមានទំហំ 4byte មានតំលៃពី 3.4×10^{-38} ដល់ 3.4×10^{38} ។
- double: សំរាប់ផ្ទុកនូវទសភាគ មានទំហំ 8byte មានតំលៃពី 1.7×10^{-308} ដល់ 1.7×10^{308} ។
- long double: សំរាប់ផ្ទុកនូវទសភាគមានទំហំ 16byte មានតំលៃពី 3.4×10^{-4932} ដល់ 3.4×10^{4932} ។

EX:

- `#include <stdio.h>`
- `int main(void)`
- `{`
- `int salary;`
- `salary = 10000;`
- `printf("My salary is %d.", salary);`
- `return 0;`
- `}`

I. Data type (Continue)

A. Identifiers and Keyword

- Identifiers: ជាអត្តសញ្ញាណសំរាប់កំណត់ឈ្មោះអថេរ ឈ្មោះអនុគមន៍ ឬឈ្មោះ Class ដែលប្រើប្រាស់នៅក្នុងការសរសេរ Code។ ដើម្បីកំណត់អត្តសញ្ញាណយើងប្រើប្រាស់គំរូដូចខាងក្រោម៖
 - ប្រើប្រាស់តួអក្សរ a-z, A-Z លេខ 1-9 សញ្ញា Underscore (_)។
 - តួអក្សរ a ខុសគ្នាពី តួអក្សរ A។
 - មិនត្រូវប្រើប្រាស់តួអក្សរពិសេសដូចជា {}, [], space, () ។
- Keyword: ពាក្យគន្លឹះជាឈ្មោះដែលមានស្រាប់នៅក្នុងភាសា C ដូចច្នេះយើងមិនអាចប្រើប្រាស់វាធ្វើជាអត្តសញ្ញាណបានទេ ព្រោះ Keyword ទាំងនោះមានអត្ថន័យ និងការប្រើប្រាស់ផ្ទាល់ខ្លួនរបស់វាផ្សេងៗ។ ជាធម្មតា Keyword សរសេរជាអក្សរតូច Small letter។ Keyword ទាំងនោះរួមមាន៖

I. Data type (Continue)

<u>auto</u>	<u>break</u>	<u>case</u>	<u>char</u>
<u>const</u>	<u>continue</u>	<u>default</u>	<u>do</u>
<u>double</u>	<u>else</u>	<u>enum</u>	<u>extern</u>
<u>float</u>	<u>for</u>	<u>goto</u>	<u>if</u>
<u>int</u>	<u>long</u>	<u>register</u>	<u>return</u>
<u>short</u>	<u>signed</u>	<u>sizeof</u>	<u>static</u>
<u>struct</u>	<u>switch</u>	<u>typedef</u>	<u>union</u>
<u>unsigned</u>	<u>void</u>	<u>volatile</u>	<u>while</u>

I. Data type (Continue)

A. Constants: Constants ជាតំលៃថេរសំរាប់ផ្តល់ឲ្យទៅអថេរដើម្បីប្រើប្រាស់ធ្វើការគណនា បង្ហាញ ឬការងារផ្សេងទៀត។

- String constants
- Numeric constants
- Character constants

1. String: String constants គឺជាក្រុមនៃតួអក្សរដែលផ្ដើមនិងបញ្ចប់ដោយសញ្ញា double quote (" ") ។

Ex: "Hello world", "How do you do?"

2. Numeric constants តំលៃថេរជាលេខ ដែលលេខទាំងនោះមាន $PI = 3.14$

3. Character constants: មានតួអក្សរតែមួយគត់ស្ថិតនៅចន្លោះ Single quote ហៅថា Character constants 'a' ។

I. ការប្រកាសប្រភេទទិន្នន័យ Variable Declaration

1. **Declaration:** ជាការប្រើប្រាស់ Variable ក្នុង C ,បើយើងចង់ប្រើប្រាស់ទិន្នន័យណាមួយចាំបាច់យើងត្រូវតែប្រកាសវាមុន។

ឧទាហរណ៍:

- char Ch1;
- int num1, num2;
- char Ch1='A';
- char Ch2 = "I love Programming";
- int num1 = 30, num2 = 45;

Syntax ក្នុងការប្រកាស:

- ***data type identifier;***
- ***data type identifier1, identifier2, identifier3...;***
- ***data type identifier=value;***
- ***data type identifier=value, identifier2=value2, identifier3=value3;***

Example:

```
#include<stdio.h>

int main( )

{
    printf( "A=10" );
    printf( "B=20" );
    return 0;
}
```

Output : A =10 B=20

```
#include<stdio.h>

int main( )
{
    int a,b;
    a=10;
    b=20;
    printf( "A=%d B=%d",a,b );
    return 0;
}
```

Output : A =10 B=20

Special Characters:

- ❖ Special Characters: នៅក្នុងភាសា C មានតួអក្សរពិសេសមួយចំនួនដើម្បីប្រើប្រាស់ទៅតាមតំរូវការអត្ថន័យ ។

Special Characters	Meanings
\a	Alert a bell character
\n	New line
\t	Horizontal tab
\b	Backspace
\r	Carriage return
\f	Form feed
\v	Vertical tab
\\	Back slash
\'	Single quote
\0	Null character
\?	Question mark

- `#include <stdio.h>`
- `int main(void)`
- `{`
- `printf("Hi there!\n\n\nThis program is a bit");`
- `printf(" longer than the others.");`
- `printf("\nBut really it's only more text.\n\n\n\\a\\a");`
- `printf("Hey, wait a minute!! What was that ? ? ?\n\n");`
- `printf("\t1.\tA bird ?\n");`
- `printf("\t2.\tA plane ?\n");`
- `printf("\t3.\tA control character ?\n");`
- `printf("\n\t\t\b\bAnd how will this look when it prints out ?\n\n");`
- `return 0;`
- `}`

II. Comments

- Comments គឺជាឃ្លាប្រយោគសំរាប់ធ្វើការអធិប្បាយទៅលើ កូដ ។ ពួកវាមិនត្រូវបាន Compile ទេ ។ ហើយវាត្រូវបានគេកំណត់ឡើង ដោយប្រើប្រាស់ tokens // ឬ /* */ ។ Token // សំរាប់បិទ មួយបន្ទាត់ រីឯ /* */ សំរាប់បិទច្រើនបន្ទាត់ (statement) ។
- Ex:

```
/* Comment for one line. Modifying r and displaying  
value of x. */  
r=100;  
printf( "x=%d\n",x );
```

Data input and output

- អនុគមន៍ printf() :
- printf() : ជា function សម្រាប់ print ទិន្នន័យដែលនៅក្នុងសញ្ញា "....." មកលើ screen

```
#include<stdio.h>

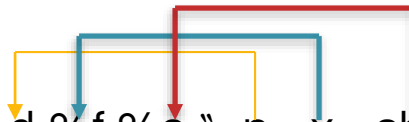
int main( )
{
    printf( "A=10" );
    return 0;
}
```

```
#include<stdio.h>

int main( )
{
    int a=10;
    printf( "A=%d ",a );
    return 0;
}
```

EX:

- `#include<stdio.h>`
- `main()`
- `{`
- `int n;`
- `float x;`
- `char ch;`
- `n= 5 ; x = 8.2 ; ch='a';`
- `printf("print values : %d %f %c ", n , x , ch);`
- `getch();`
- `return(0);`
- `}`



Format Control

int	%d
Long int	%ld
float	%f
double	%lf
Char	%c
String	%s

Variable Syntax : Datatype Variable_name;

Output Syntax : printf("format Control",Variable);

Input Syntax : printf("format Control",&Variable);

scanf() function

- scanf() សម្រាប់ចាប់ទិន្នន័យពី keyboard
- #include <stdio.h>
- main()
- {int n,m,s;
- printf("enter n= ");
- scanf("%d",&n);
- printf("enter m= ");
- scanf("%d",&m);
- S=n+m;
- printf("sum values m and n =%d",s);
- return(0);
- }

EX:

- `#include<stdio.h>`
- `#include<conio.h>`
- `main()`
- `{`
- `char name[50];`
- `printf("Enter your name :");`
- `gets (name);`
- `printf("\n your name is %s ", name);`
- `getch();`
- `return (0);`
- `}`

II. Operators

a. Arithmetic Operators ប្រមាណវិធីពីជគណិតប្រើប្រាស់សំរាប់បង្ហាត់ឲ្យកម្មវិធីធ្វើការគណនាពីរចំនួន ឬច្រើនជាងនេះ។ សូមមើលប្រមាណវិធី ពីជគណិត និងអត្ថន័យរបស់វាដូចខាងក្រោម៖

Operators	Meanings	Example
+	Addition	23+8
-	Subtraction	x-y
*	Multiplication	2*a
/	Division	N/230
%	Modulo	N%2

ប្រមាណវិធីពីជគណិត +, -, *, and / មានការប្រើប្រាស់ដូចគ្នាទៅនឹងការប្រើប្រាស់នៅក្នុងគណិតវិទ្យាដែរ មានប្រមាណវិធីតែមួយគត់ដែលយើងពុំសូវជួបប្រទះនៅក្នុងការប្រើប្រាស់គឺ % (Modulo)។ ដែល Modulo គឺជាប្រមាណវិធីសំរាប់ចែករកសំណល់។ Ex: $x=11\%3$;

EX:

- #include <stdio.h>
- main()

- {
- int a , b ;
- float x,y ;
- a = 10 ;b = 3 ;
- printf("a=%d \nb=%d",a , b);
- printf("\na-b=%d",a-b);
- printf("\na+b = %d ", a+b);
- printf("\na * b= %d",a * b);
- x=a/b;
- printf("\na/b= %f",x);
- y=a%b;
- printf("\na%b=%.f",y);
- return (0);
- }

Operators (continue)

b. Assignment Operators: សំរាប់ផ្តល់តំលៃទៅអោយអថេរ ផ្តល់តំលៃពីខាងស្តាំទៅខាងឆ្វេង។ Ex: a=5;

Operators	Meanings	Example
=	To assign the value from the right to the left	X=45

c. Compound Assignment Operators: នៅពេលយើងចង់ធ្វើការកែប្រែតំលៃនៃអថេរមួយដោយប្រើប្រាស់ប្រមាណវិធីនៅលើតំលៃចាស់នៃអថេរនោះ យើងប្រើប្រាស់ Compound Assignment Operator ។

Operators (continue)

Operators	Meanings	Example
+=	To add the value of the left variable with the value on the right then assign back to the variable.	X+=45
-=	To subtract the value of the left variable with the value on the right then assign back to the variable.	X-=45
=	To multiple the value of the left variable with the value on right then assign back to the variable.	X=3
/=	To divide the value of the left variable with the value on the right then assign back to the left.	X/=45
%=	To modulo the value of the left variable with the value on the right and assign back to the left.	X%=45

Operators (continue)

- d. Comparison Operators: យើងប្រើប្រាស់ប្រមាណវិធីប្រៀបធៀបនៅពេលដែលយើងធ្វើការប្រៀបធៀបរវាងពីរចំនួន។

Operators	Meanings
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
==	Equal to
!=	Not Equal to

ឧទាហរណ៍

`(7 == 5) // evaluates to false.`

`(5 > 4) // evaluates to true.`

`(3 != 2) // evaluates to true.`

Operators (continue)

- $(6 \geq 6)$ // evaluates to true.
- $(5 < 5)$ // evaluates to false.
- តាមឧទាហរណ៍ខាងលើយើងប្រើប្រាស់ចំនួនថេរជាលេខសំរាប់ធ្វើការប្រៀបធៀប យើងក៏អាចនូវកន្សោមត្រឹមត្រូវមួយបានដែរ រួមទាំងអថេរជាដើម។
- ឧទាហរណ៍ សន្មតថា $a=2$, $b=3$ និង $c=6$
- $(a == 5)$ // evaluates to false since a is not equal to 5.
- $(a * b \geq c)$ // evaluates to true since $(2 * 3 \geq 6)$ is true.
- $(b+4 > a * c)$ // evaluates to false since $(3+4 > 2 * 6)$ is false.
- $((b=2) == a)$ // evaluates to true.
- **ចំណាំ៖** គួរប្រុងប្រយ័ត្នរវាងសញ្ញាប្រមាណវិធី $=$ និង $==$ ។ $=$ គឺសំរាប់ផ្តល់តំលៃទៅខាងធ្វេង រីឯ $==$ សំរាប់ធ្វើការប្រៀបធៀប ពិតនៅពេលអង្គទាំងពីរស្មើគ្នា។

Operators (continue)

- e. Logical Operators:

Operators	Meanings
&&	Logical And
	Logical Or
!	Logical Not

ការប្រើប្រាស់ !

!(5 == 5) // evaluates to false because the expression at its right (5 == 5) is true.

!(6 <= 4) // evaluates to true because (6 <= 4) would be false.

!true // evaluates to false

!false // evaluates to true.

Operators (continue)

- ការប្រើប្រាស់ && សន្មត់ថាមានអថេរ a និង b មានតំលៃដូចក្នុងតារាង

a	b	a && b
true	true	true
true	false	false
false	true	false
false	false	false

ការប្រើប្រាស់ || សន្មត់ថាមានអថេរ a និង b មានតំលៃដូចក្នុងតារាង

a	b	a b
true	true	true
true	false	true
false	true	true
false	false	false

Operators (continue)

- ឧទាហរណ៍
- `((5 == 5) && (3 > 6)) // evaluates to false (true && false)`.
- `((5 == 5) || (3 > 6)) // evaluates to true (true || false)`.

f. Condition operator: Condition operator (?) ធ្វើការវាយតម្លៃកន្សោមរួចផ្តល់តម្លៃលទ្ធផលទីមួយប្រសិនបើលក្ខណៈនៃកន្សោមពិត និងផ្តល់លទ្ធផលទីពីរផ្ទុយមកវិញ។

Syntax:

`condition ? result1 : result2`

ប្រសិនបើ condition ពិត កន្សោមទទួលបានលទ្ធផល result1 ហើយផ្ទុយមកវិញទទួលបានលទ្ធផល result2



Operators (continue)

Operators (continue)

- f. Special Operators: មានប្រមាណវិធីពិសេសមួយចំនួននៅក្នុង C សំរាប់ការប្រើប្រាស់ជាពិសេស។ ដែលប្រមាណវិធីទាំងនោះត្រូវបានគេហៅថា Unary Operator ដែលការប្រើប្រាស់ត្រូវការ អង្គតែមួយប៉ុណ្ណោះ។

Operators	Meanings
*	Content of storage field to which a pointer is pointing
&	Address of a variable
-	Negative value

Operators (continue)

f. Increase and Decrease

Operators	Meanings
++	Increment
--	Decrement

++ ឬ -- បន្ថែមនិងបន្ថយ ១ ទៅលើតំលៃរបស់អថេរ។ វាមានអន្លាយស្មើនឹង +=1 និង -=1។

ឧទាហរណ៍៖

```
c++;
```

```
c+=1;
```

```
c=c+1;
```

Statement ទាំងបីមានន័យដូចគ្នាគឺ បន្ថែម ១ ទៅលើតំលៃរបស់អថេរ c។

ចំណាំ៖ ប្រមាណវិធីទាំងពីរនេះអាចប្រើប្រាស់បាននៅពីខាងមុខ Prefix និងខាងក្រោយ Postfix នៃអថេរ មានន័យថាអាចសរសេរបាន(++a) ឬ(a++)។ នៅក្នុងករណីនេះ ++a តំលៃរបស់អថេរគឺកើនឡើងមុនពេលកន្សោមទទួលយកតំលៃ រីឯ a++ តំលៃរបស់អថេរគឺកើនឡើងក្រោយពេលកន្សោមទទួលយកតំលៃ។

Operators (continue)

- ឧទាហរណ៍១៖
- $B=3;$
- $A=++B;$ // A contains 4, B contains 4
- ឧទាហរណ៍២៖
- $B=3;$
- $A=B++;$ // A contains 3, B contains 4

លំហាត់អនុវត្ត

បំហាត់អនុវត្ត

លំហាត់អនុវត្ត

បំហាត់អនុវត្ត

Home work

☞ ចូរសរសេរកម្មវិធីដែលមានមួយដែលមានលទ្ធភាពគណនារក បរិមាត្រនៃរង្វង់ និង ក្រលាផ្ទៃនៃរង្វង់។ នៅពេលដំណើរការកម្មវិធីអនុញ្ញាតិអោយអ្នកប្រើប្រាស់បញ្ចូលលេខប្រវែងនៃកាំរង្វង់ បន្ទាប់មកចុច Enter ពេលនោះកម្មវិធីនឹងបង្ហាញលទ្ធផល

ឧទាហរណ៍៖

Circumference is: 18.84 meter (បរិមាត្ររង្វង់)

Circle area is: 28.27 square meter (ក្រលាផ្ទៃរង្វង់)

ចូរសរសេរកម្មវិធីដើម្បីបញ្ចូល ឈ្មោះ , ភេទ, អាយុ, ព្រមទាំង Score លើមុខវិជ្ជាដូចជា៖ គណិត រូប គីមី រួច ហើយរកពិន្ទុសរុប រកមធ្យមភាគ និងចំណាត់ថ្នាក់ ?

Name		Gender	
Age	Score	Avg	
LONG Dara	Male	20	90

A

ក្នុងលក្ខខណ្ឌ

បើ $\text{Score} > 90 \ \&\& \ \text{Score} \leq 100$

បានចំណាត់ថ្នាក់ A

បើ $\text{Score} > 80 \ \&\& \ \text{Score} < 90$

បានចំណាត់ថ្នាក់ B

បើ $\text{Score} > 70 \ \&\& \ \text{Score} < 80$

បានចំណាត់ថ្នាក់ C

បើ $\text{Score} > 60 \ \&\& \ \text{Score} < 70$

បានចំណាត់ថ្នាក់ D

បើ $\text{Score} \geq 50$

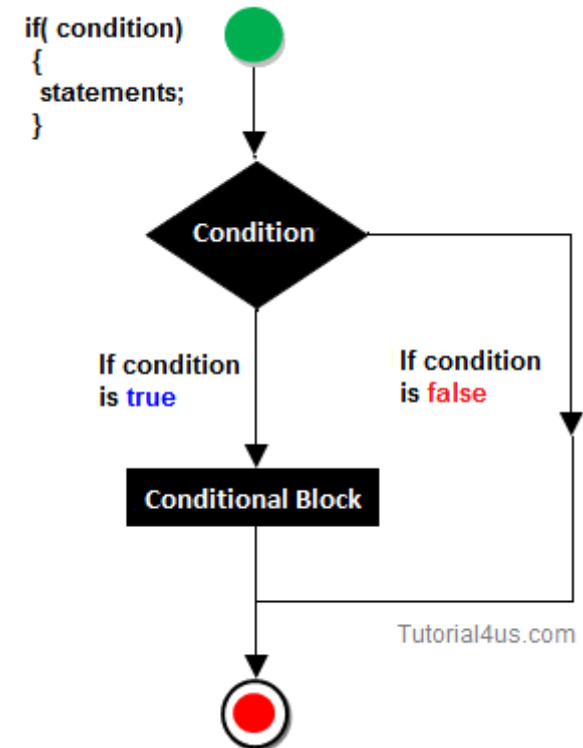
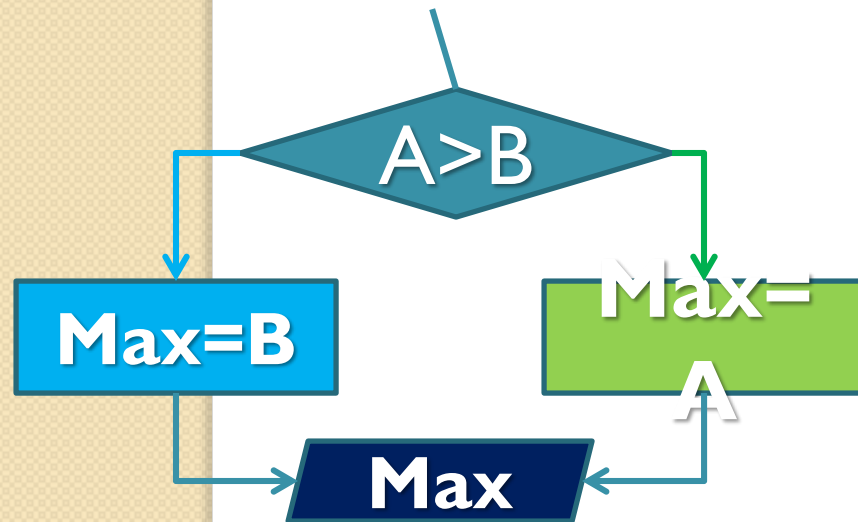
បានចំណាត់ថ្នាក់ E

ក្រៅពីនេះ

បានចំណាត់ថ្នាក់ F

Making Decisions

មេរៀនទី ៣



ក្នុងមេរៀននេះយើងនឹងរៀន

I. Condition Statement

a. If Statement

b. If.....Else
Statement

c. Nested If
Statement

II. Switch Statement

I. Conditional Statement

Condition Statement ឬ Conditional Expressions គឺត្រូវបានប្រើដើម្បីត្រួតពិនិត្យលក្ខខណ្ឌ ហើយនិង កំណត់ទិសដៅក្នុងគោលបំណងដែលយើងចង់ធ្វើអ្វីមួយ។

a. *if* Statement

Syntax:

```
if( Expression) statement;
```

```
if( Expression)
```

```
{
```

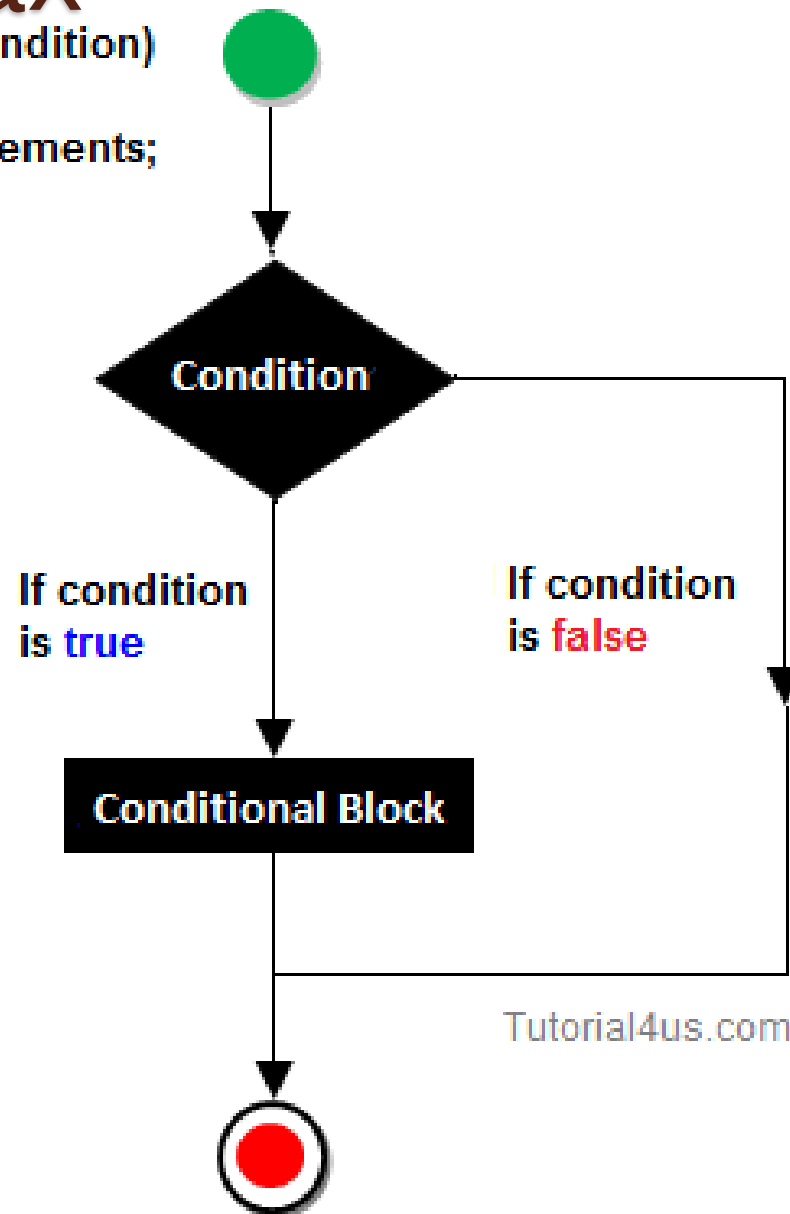
```
    Statements;
```

```
}
```

-*Expression* គឺជាលក្ខខណ្ឌដើម្បីត្រួតពិនិត្យ *if* ពិត រឺ មិនពិត

Syntax

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



Tutorial4us.com

```
#include<stdio.h>
#include<conio.h>
void main()
```

```
{
```

```
    int age=40;
```

```
    if(age<18)
```

```
    {
```

```
        printf("you are child");
```

```
    }
```

```
    else
```

```
    {
```

```
        printf("you are young");
```

```
    }
```

```
    return (0);
```

```
}
```

EXAMPLE1: ប្រើប្រាស់ IF.....STATEMENT

```
1.  #include<stdio.h>
2.  #include<conio.h>
3.  void main( )
4.  {
5.      int a=30,b=20;
6.      if( a>b ) printf( "a is greater b" );
7.      getch( );
8.  }
```

យើងប្រើ
if ...statement ដើម្បីប្រៀប
ធៀបតម្លៃពីរ។

ជឿនលឿន

B. កម្រិត IF....ELSE

STATEMENT

❖ Syntax:

if (*expression*)

statement;

else

statement;

if (*expression*)

{

statement(s);

}

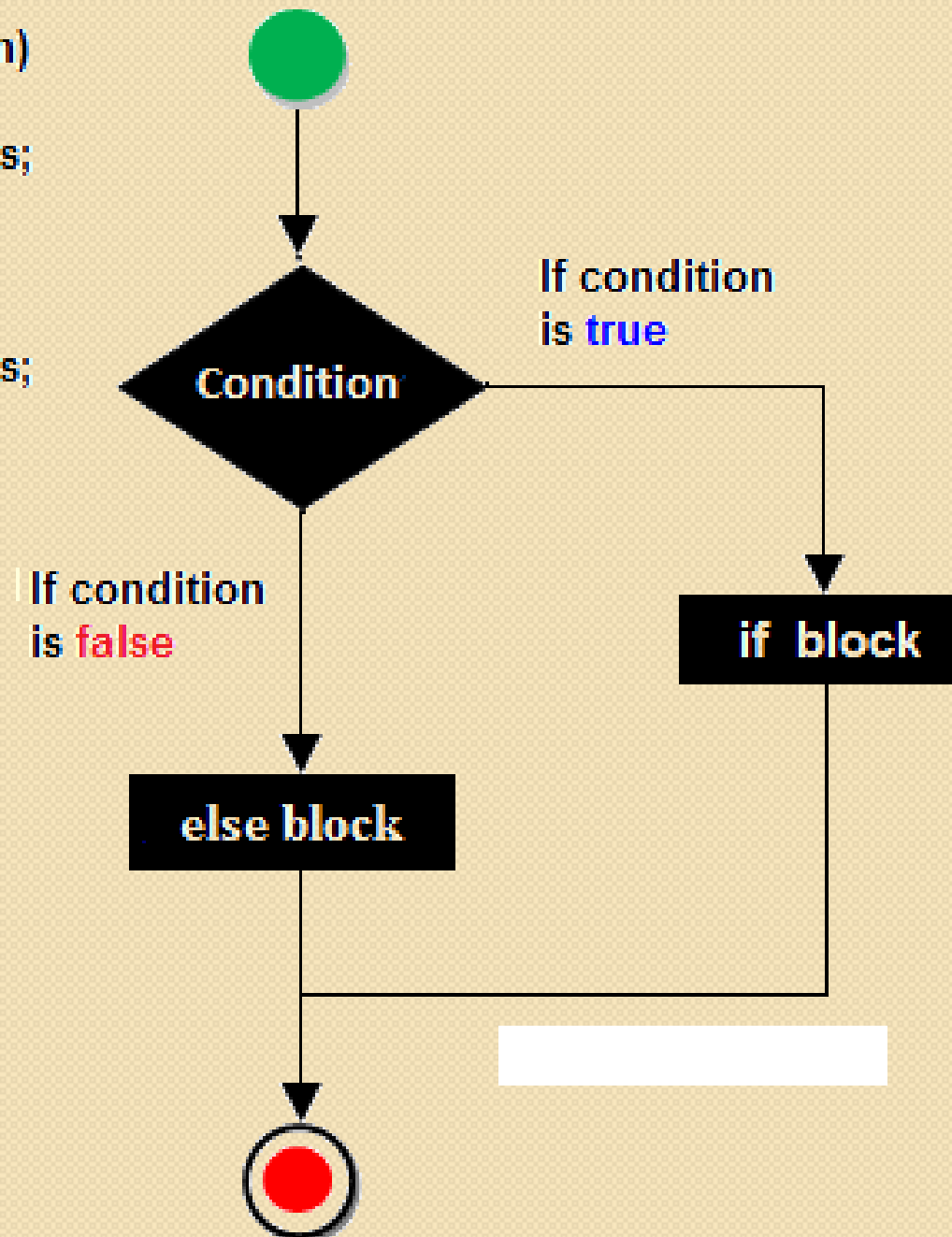
else

{

statement(s);

}

```
if( condition)
{
    statements;
}
else
{
    statements;
}
```



EX: Odd and Even Number

- `#include <stdio.h>`
- `main()`
- `{`
- `int num ;`
- `printf("Check Number Odd and Even:\n");`
- `printf("Enter Number:");`
- `scanf("%d",&num);`
- `if(num%2==0)`
 - `printf("Number is Even");`
- `else`
 - `printf("Number is Odd");`
- `return ;`
- `}`

ឧទាហរណ៍ ពីការប្រើប្រាស់ *if ...else* statement

```
• #include <stdio.h>
• #include <conio.h>
• main( )
• {
•     int n1 ,n2 ,min , max;
•     printf( "Enter the first Value=:");
•     scanf ( "%d", &n1 );
•     printf ( "enter the second value:" );
•     scanf ( "%d", &n2 );
•     if ( n1<n2 )
•         min = n1;
•         max = n2;
•     else
•         min = n2;
•         max = n1;
•     printf ( "\n maximum= %d ",max );
•     printf ( "\n minimum= %d ",min );
•     getch( );
• }
```

C.ការប្រើប្រាស់ *IF.....ELSE IF* STATEMENT

❖ចំណាំ៖

ត្រូវបានយើងប្រើដើម្បីត្រួតពិនិត្យលក្ខខណ្ឌច្រើនគៗគ្នា

❖Syntax:

if(expression)

statement(s);

else if (expression)

statement(s);

else

ដោះស្រាយសមីការ $ax + b > 0$

- `#include<stdio.h>`
- `#include<conio.h>`
- `#include<math.h>`
- `int main()`
- `{`
- `float a,b,x;`
- `printf("Enter a=");`
- `scanf("%f",&a);`
- `printf("Enter b=");`
- `scanf("%f",&b);`
- `if(a==0)`
- `{`

```
if(b>0)
printf("\n Infinitive");
else
printf("\n No root");
}
if(a>0)

printf("\n x> %f ",(-b/a));
if(a<0)
printf ("\nx< %f",(-b/a));
getch();
return(0);
}
```

Homework

ចូរសរសេរកម្មវិធីមួយដើម្បីឱ្យ Users បញ្ចូល Name (មានដកឃ្លា) , Gender , Age, ព្រមទាំង Score។
ដោយឱ្យបញ្ហាញមកវិញនូវទំរង់ដូចខាងក្រោម៖

Name	Gender	Age	Score	Avg
LONG Dara	Male	20	90	A

ក្នុងលក្ខខណ្ឌ

បើ $\text{Score} > 90 \ \&\& \ \text{Score} \leq 100$

បានចំណាត់ថ្នាក់ A

បើ $\text{Score} > 80 \ \&\& \ \text{Score} < 90$

បានចំណាត់ថ្នាក់ B

បើ $\text{Score} > 70 \ \&\& \ \text{Score} < 80$

បានចំណាត់ថ្នាក់ C

បើ $\text{Score} > 60 \ \&\& \ \text{Score} < 70$


បានចំណាត់ថ្នាក់ D

បើ $\text{Score} \geq 50$

បានចំណាត់ថ្នាក់ E

ក្រៅពីនេះ

បានចំណាត់ថ្នាក់ F



```
#include<stdio.h>
```

```
int main( )
```

```
{
```

```
    int age;
```

```
    char name[20],sex[20];
```

```
    float m,c,e,total,aver;
```

```
    printf( "input name:" );scanf( "%s",&name );
```

```
    printf( "input sex:" );scanf( "%s",&sex );
```

```
    printf( "input age:" );scanf( "%d",&age );
```

```
    printf( "input m:" );scanf( "%f",&m );
```

```
    printf( "input c:" );scanf( "%f",&c );
```

```
    printf( "input e:" );scanf( "%f",&e );
```

```
    total=c+m+e;
```

```
    printf( "\ntotal=%f",total );
```

```
    aver=total/3;
```

```
    printf( "\naver=%f",aver );
```



○ if(aver>=90)

printf("\nyou are grade A");

else if(aver>=80)

printf("\nyou are grade B");

else if(aver>=70)

printf("\nyou are grade C");

else if(aver>=60)

printf("\nyou are grade D");

else if(aver>=50)

printf("\nyou are grade E");

else

printf("\nyou are grade F");

printf("\n\nName\tSex\tAge\tScore\n");

printf("\n%s\t%s\t%d\tScore\n",name,sex,age);


}

Homework $ax^2+bx+c=0$

- `#include <stdio.h>`
- `#include <math.h>`
- `main()`
- `{`
- `float a , b ,c ;`
- `float x1 , x2 , delta;`
- `printf("\n a = "); scanf("%f",&a);`
- `printf("\n b = "); scanf("%f",&b);`
- `printf("\n c = "); scanf("%f",&c);`
- `delta = (b * b)-(4 * a * c);`
- `printf("\n delta = %.2f\n",delta);`
- `}`

- if (delta==0)
- {
- printf("\n Only One Root");
- printf("\n x = %.2f",-b/(2 *a));
- }
-

- `if(delta>0)`
- `{`
- `printf("\n Two Root");`
- `x1=(-b+sqrt(delta))/(2 *a);`
- `x2=(-b-sqrt(delta))/(2 *a);`
- `printf("\n X1 = %.2f",x1);`
- `printf("\n X2 = %.2f",x2);`
- `}`
-

- 
- `if (delta<0)`
 - `printf("No root");`
 - `getch();`
 - `return (0);`
 - `}`

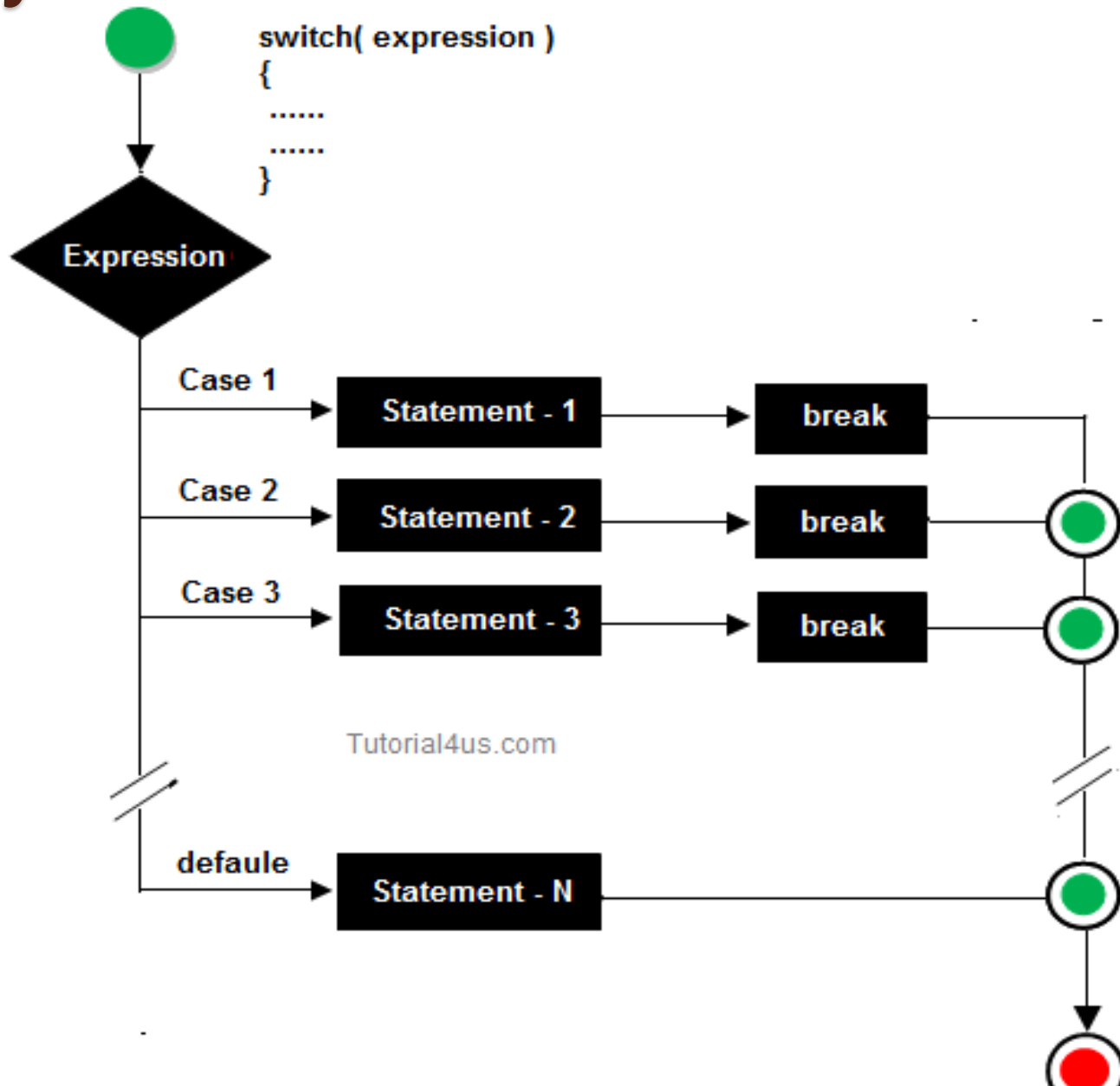
II. Switch Statement ការប្រើ Switch

- ឃ្លា Switch គឺជាឃ្លាពិសេសដែលអាចឲ្យគេប្រតិបត្តតាម តម្លៃណាមួយដែលមានលក្ខខណ្ឌត្រូវគ្នា ក្នុងចំណោមតម្លៃជាច្រើន។
- តម្លៃរបស់វា បើ ហើយ វាមានទម្រង់ដូចខាងក្រោម៖
- `switch (expression)`
 - `Case constant1 :`
 - Statement ; `break`;
 - `Case constant 2 :`
 - Statement ; `break`;
 -
 - `Case constant N :`
 - Statement ; `break`;
 - `default :`
 - Statement ; `break`;
 - }

Note:

- `break` គឺអាចអត់ដាក់ក៏បាន

Symtax



ឧទាហរណ៍១: ចូរពិនិត្យមើល code នៃកម្មវិធីខាងក្រោម

- Switch (n);
- {
- Case 0 : printf("\n Number Zero ") ; break ;
- Case 1 : printf("\n Number One ") ; break ;
- Case 2 : printf("\n Number Two ") ; break ;
- }
- យើងឃើញថា code ខាងលើនេះ បើសិនជាវាផ្ទៀងផ្ទាត់នឹងលក្ខខណ្ឌ ខ្លះមួយនោះវានឹងបង្ហាញតម្លៃដែលវាផ្ទៀងផ្ទាត់រហូតមកដល់ក្រោម ។ ដូចនេះដើម្បីផ្តាច់លក្ខណ៍ពី case មួយទៅ case មួយទៀតយើង គួរប្រើឃ្លា break ។

III. Breaking Control Statement

❖ **break** : ត្រូវបានគេប្រើប្រាស់ក្នុងរបៀបពីរយ៉ាងគឺ :

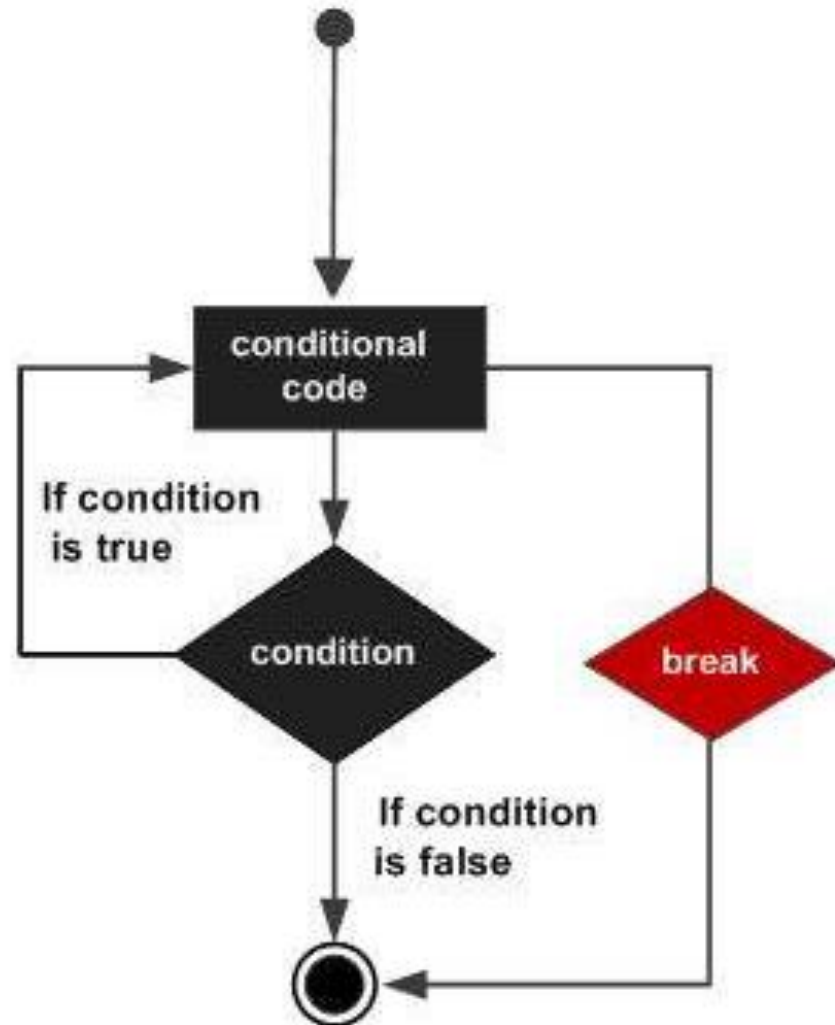
- គេប្រើប្រាស់ក្នុង loop ដើម្បីបញ្ឈប់ដំណើរការរបស់ loop ក្នុងលក្ខខណ្ឌណាមួយ ក្រោយមកបន្តទៅដំណើរការនូវ statements ផ្សេងទៀត។
- វាអាចប្រើដើម្បីបញ្ឈប់ case នៅក្នុង **switch** statement

រូបមន្តរបស់វាគឺ៖

break;

ចូរពិនិត្យមើល Diagram ដំណើរការខាង ក្រោម៖

Flow Diagram:



```
#include<stdio.h>
#include<conio.h>
void main() {
    int ch;
    printf("Enter any number (1 to 7)");
    scanf("%d",&ch);
    switch(ch)
    {
        case 1: printf("Today is Monday"); break;
        case 2: printf("Today is Tuesday"); break;
        case 3: printf("Today is Wednesday"); break;
        case 4: printf("Today is Thursday"); break;
        case 5: printf("Today is Friday"); break;
        case 6: printf("Today is Saturday"); break;
        case 7: printf("Today is Sunday"); break;
        default:
            printf("Only enter value 1 to 7");
    }
    getch();
    Return 0 ;
}
```

Homework

```
#include <stdio.h>
#include <conio.h>
void main()
{ char choice;
  int a,b,res=0;
  clrscr();
  printf("Enter first value: ");
  scanf("%d",&a);
  printf("\n Enter operator: ");
  choice=getch();
  printf("\n Enter second value: ");
  scanf("%d",&b);
```

```
switch(choice)
{ case '+': res=a+b;
  printf("Sum: %d",res); break;
  case '-':
  res=a-b; printf("Minus: %d",res);
  break;
  case '*':
  res=a*b; printf("Product: %d",res);
  break;
  case '/':
  res=a/b; printf("Divide: %d",res); break;
  default: printf("Enter Valid
Operator!!");
}
getch();
}
```

```
#include <stdio.h>
int main(void)
{
    char answer = 0; // Stores an input character
    printf("Enter Y or N: ");
    scanf(" %c", &answer);
    switch(answer)
    {
        case 'y': case 'Y':
            printf("You responded in the affirmative.\n");
            break;
        case 'n': case 'N':
            printf("You responded in the negative.\n");
            break;
        default:
            printf("You did not respond correctly. . .\n");
            break;
    }
    return 0;
}
```




មេរៀនទី៤

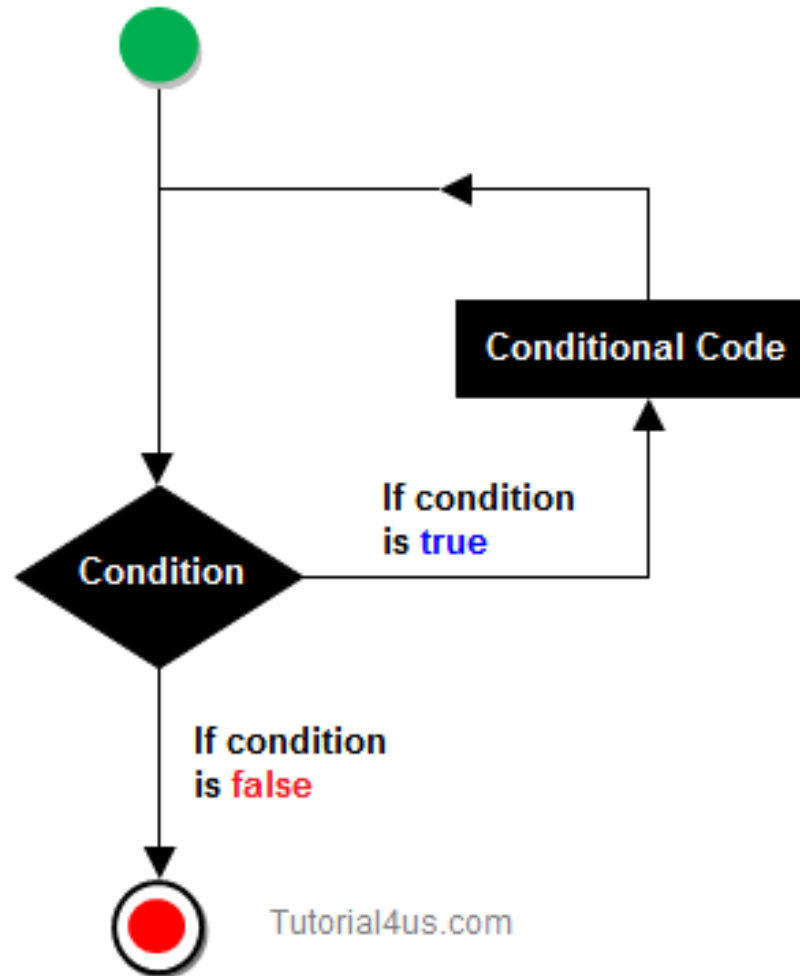
Loop Control Structure

ប្រភេទ Loops នៅក្នុងភាសា c

Loop ជាដំណើរការដដែលៗដោយគោរពតាមលក្ខណៈ។ ក្នុងភាសា c Loop មានបីប្រភេទ៖

- for loop
- while loop
- do...while

loop

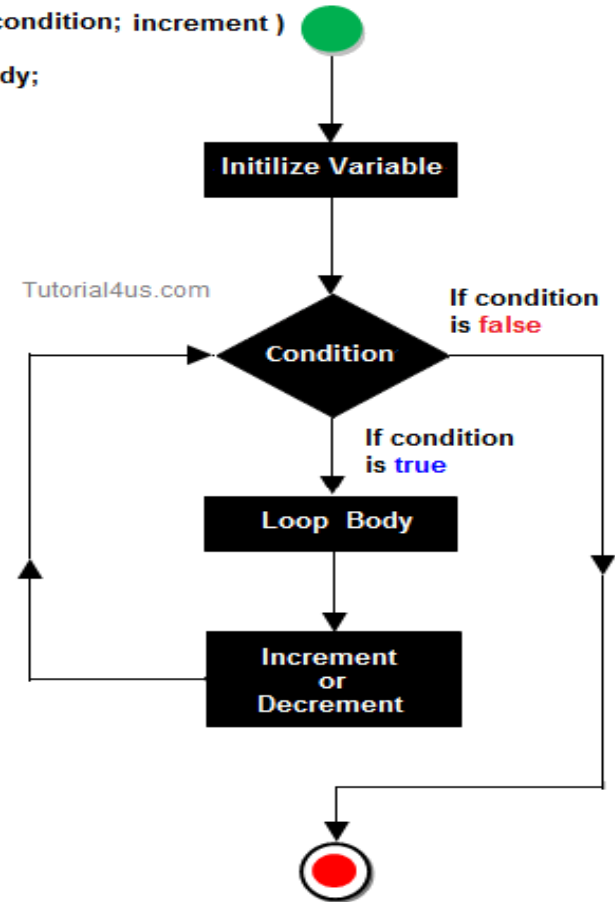


For loop

for loop រួមមានបីកន្សោម និងក្នុងតួខ្លួនរបស់ For អាចមានជា statemnets ដែលសរសេរខាងក្រោម ។
ចំពោះកន្សោមទាំងបីអាចគ្មានកន្សោមណាមួយបាន តែត្រូវមានសញ្ញា (;) ដើម្បី ខណ្ឌចែកពីកន្សោមមួយ ទៅកន្សោមមួយទៀតជាដាច់ខាត។

- Initialization: ចាប់ផ្តើមអោយតម្លៃ
- Condition : លក្ខណ្ឌ
- Increment or Decrements: កើនឡើង ឬ ថយ

```
for( init; condition; increment )  
{  
    loop body;  
}
```



EX:

```
#include<stdio.h>
#include<conio.h>
main()
{
int i;
for(i=1;i<10;i++)
{
printf("\nHello NTTI");
}
getch();
return 0;
}
```

```
#include<stdio.h>
#include<conio.h>
main()
{
int i;
for(i=1;i<=10;i++)
{
printf("\n%d",i);
}
getch();
return 0;
}
```

```
#include<stdio.h>
#include<conio.h>
main()
{
int i , sum=0;
for(i=1;i<10;i++)
{
printf("%d\n",i);
sum= sum + i ;
}
printf("\n-----");
printf("\nSum=%d",sum);
getch();
return 0;
}
```

```
#include <stdio.h>
```

```
main()
```

```
{
```

```
int j,n;
```

```
printf("Input the number (Table to be calculated) : ");
```

```
scanf("%d",&n);
```

```
printf("\n");
```

```
for(j=1;j<=10;j++)
```

```
{
```

```
    printf("%d X %d = %d \n",n,j,n*j);
```

```
}
```

```
}
```

Sum of even number and odd number

```
#include <stdio.h>
main()
{
    int i,n,sum=0;
    for(i=0;i<=10;i++)
    {
        if ( i%2==0)
        {
            printf("%d ",i);
            sum+=i;
        }
    }
    printf("\nSum of even number : %d \n",sum);
}
```

```
#include <stdio.h>
main()
{
    int i,n,sum=0;
    for(i=0;i<=10;i++)
    {
        if ( i%2!=0)
        {
            printf("%d ",i);
            sum+=i;
        }
    }
    printf("\nSum of odd number : %d \n",sum);
}
```


EX2

```
#include <stdio.h>
main()
{
    int i,n,sum=0;
        for(i=0;i<=n;i=i+2)
        {
            printf("%d ",i);
            sum+=i;
        }
    printf("\nSum of even number : %d \n",sum);
}
```

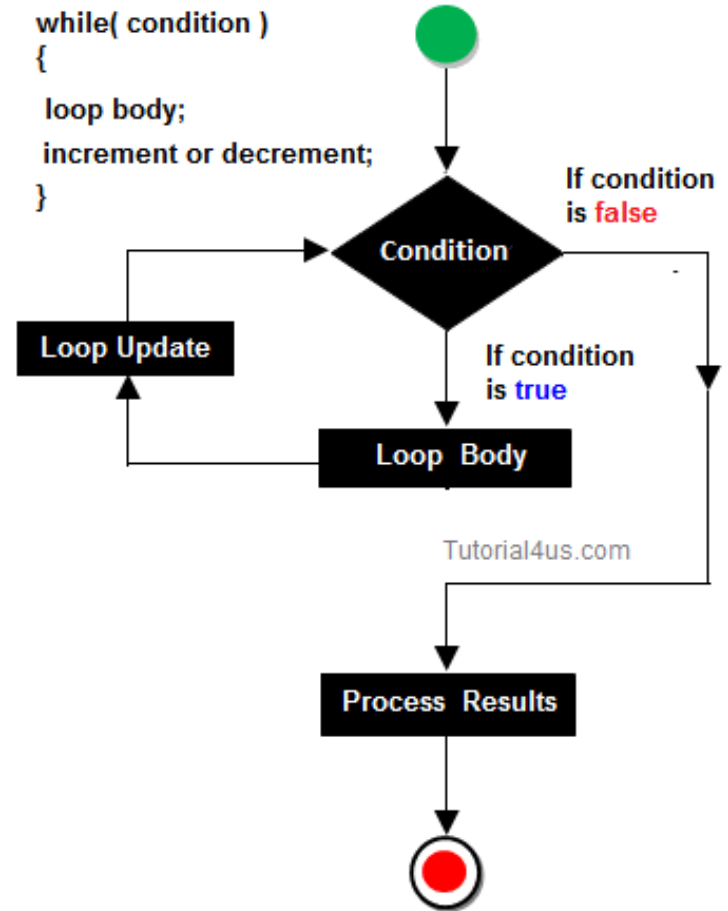
```
#include <stdio.h>
main()
{
    int i,n,sum=0;
        for(i=1;i<=n;i=i+2)
        {
            printf("%d ",i);
            sum+=i;
        }
    printf("\nSum of odd number : %d \n",sum);
}
```

```
#include <stdio.h>
main()
{
    int i,n,sum=0;
    printf("Input number of terms : ");
    scanf("%d",&n);
    printf("\nThe even numbers are :");
    for(i=0;i<=n;i++)
    {
        printf("%d ",2*i);
        sum+=2*i;
    }
    printf("\nThe Sum of even Number %d terms : %d \n",n,sum);
}
```

while

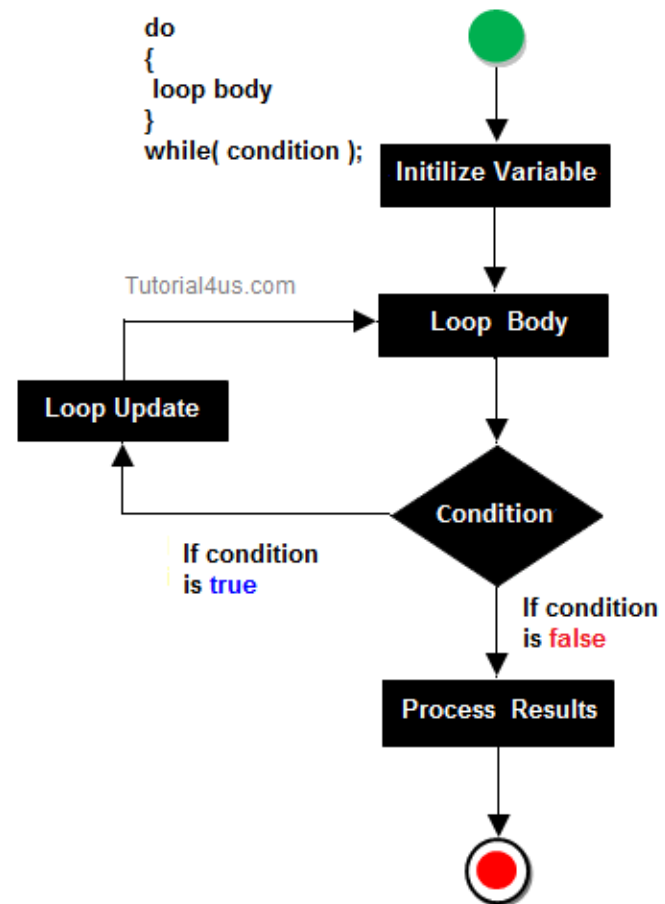
While រង្វិលជុំដំបូងពិនិត្យមើលលក្ខខណ្ឌ
ប្រសិនបើលក្ខខណ្ឌពិត
statementទៅខាងក្នុងរង្វិលជុំដទៃទៀតមា

ន។



```
#include<stdio.h>
#include<conio.h>
main()
{
    int i=1;
    while(i<5)
    {
        printf("\n%d",i);
        i++;
    }
    getch();
    return 0;
}
```

do-while



```
#include<stdio.h>
#include<conio.h>
main()
{
    int i=1;
        do
        {
            printf("\n%d",i);
            i++;
        } while(i<5);
        getch();
        return 0 ;
}
```

```
#include <stdio.h>
#include <conio.h>
main()
```

```
{
```

```
    int a , s = 0;
```

```
    do
```

```
    {
```

```
        printf("\n Enter a = ");
```

```
        scanf("%d",&a);
```

```
        s=s+a;
```

```
    }while (a!=0);
```

```
    printf("Sum = %d",s);
```

```
    getch();
```

```
    return 0 ;
```

```
}
```

Break statements

ក្នុង ករណីពេលជួប
break statement វានឹង
បញ្ឈប់ loop ហើយទៅ
អនុវត្ត statements ដែល
នៅពីក្រោយវា។

```
#include <stdio.h>
main()
{
    int i;
    for(i=1;i<=10;i++)
    {
        if ( i==5)
        {
            break;
        }
        printf("\n%d",i);
    }
    getch();
    return 0;
}
```


Continue Statements

ក្នុង ករណីពេលជួប continue statement វានឹងទៅអនុវត្ត loop statements បន្ទាប់មក ទៀតដោយមិនចាំបាច់អនុវត្ត statement ដែលនៅសល់។

```
#include <stdio.h>
main()
{
    int i;
    for(i=1;i<=10;i++)
    {
        if ( i==5)
        {
            continue;
        }
        printf("\n%d",i);
    }
    getch();
    return 0;
}
```



Nest Loop

```
#include <stdio.h>
#include <conio.h>
main()
{
    int i, j;
    for(i=1;i<=5;i++)
    {
        for(j=1;j<=i;j++)
        {
            printf("%d",i);
        }
        printf("\n");
    }
    getch();
}
```

```
#include<stdio.h>
#include<conio.h>
main()
{
    int i, j;
    for(i=1;i<=5;i++)
    {
        for(j=1;j<=5;j++)
            printf("%d",i);
        printf("\n");
    }
    getch();
}
```

```
#include<stdio.h>
#include<conio.h>
main()
{
    int i, j;
    for(i=5;i>=1;i--)
    {
        for(j=5;j>=1;j--)
        {
            printf("%d",j);
        }
        printf("\n");
    }
    getch();
}
```

```
#include<stdio.h>
#include<conio.h>
main()
{
    int i, j;
        for(i=1;i<=5;i++)
        {
            for(j=1;j<=i;j++)
            {
                printf("%c",'A' + j-1);
            }
            printf("\n");
        }
    getch();
}
```

មេរៀនទី ៥

Array

Definition


- Array គឺជាសំនុំនៃ Variable ដែលមាន Data type ដូចគ្នា និងមានឈ្មោះតែមួយ ។ ធាតុនីមួយៗរបស់ Array គឺជា Variable 1 បើមានការកែប្រែទៅលើធាតុនោះ ធាតុដទៃគ្មាន ការប្រែប្រួលឡើយ ។
- គេសំគាល់ធាតុរបស់ Array ដោយ Index ឬក៏ Subscript ដែលធាតុទី១ មាន $\text{Index} = 0$ រៀងគ្នារហូត ដល់ទី n ដែល មាន $\text{Index} = n-1$ ។
- ទំហំរបស់ Array ឬប្រវែងរបស់ Array ត្រូវតែជាចំនួនគត់ ហើយធំជាងសូន្យ ។

Syntax

- Data - Type Array_Name[Length];
- Data – Type : ជាប្រភេទ Data type របស់ Array
- Array_Name : ជាឈ្មោះរបស់ Array
- Length : ជាប្រវែងរបស់ Array

EX:

- `Int arr[5];`
- បញ្ជាក់ថាមាន 5 variables ជាប្រភេទ Integer ។
- `int` ជា Data type of Array
- `arr` ជា Name of Array
- 5 ជាប្រវែងរបស់ Array ដែលមាន Index 0,1,2,3,4



```
int arr[5];  
arr[0]=10;  
arr[1]=20;  
arr[2]=30;  
arr[3]=40;  
arr[4]=50;
```

```
int arr[]={10,20,30,40,50};
```

- `cn [0] = 5;`
- `cn [1] = 3;`
- `cn [2] = 7;`
- `cn [3] = 4;`
- `cn [4] = 2;`
- `cn [5] = 3;`
- `cn [6] = 4;`
- `cn [7] = 6;`
- `cn [8] = 1;`
- `cn [9] = 5;`
- `cn [10] = 8;`
- `cn [11] = 4;`

5	3	7	4	2	3	4	6	1	5	8	4
cn[0]	cn[1]	cn[2]	cn[3]	cn[4]	cn[5]	cn[6]	cn[7]	cn[8]	cn[9]	cn[10]	cn[11]
1 st	2 nd	3 rd	4 th	5 th	6 th	7 th	8 th	9 th	10 th	11 th	12 th

```
int cn[ 12 ] = { 5, 3, 7, 4, 2, 3, 4, 6, 1, 5, 8, 4 };
```

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int i;
    int nums[ ]={80, 62, 70, 90, 98};
        for(i=0;i<5;i++)
        {
            printf("\n%d",nums[i]);
        }
    getch();
}
```

- `# include <stdio.h>`
- `# include <conio.h>`
- `# define SIZE 5`
- `main ()`
- `{ int x [SIZE]; int i ;`
- `for (i = 0; i < SIZE ; i ++)`
- `{ printf ("x[%d] =", i); scanf ("%d",&x[i]);`
- `}`
- `printf ("\n Display Array x :");`
- `for (i = 0; i < SIZE ; i ++)`
- `{printf ("\n x[%d] = %d", i, x[i]);}`
- `getch ();`
- `return (0);`
- `}`

```
#include <stdio.h>
int main()
{
    int i, n;
    float nums[10], sum = 0, average;
    printf("Enter n: ");
    scanf("%d", &n);
    for(i=0; i<n; ++i)
    {
        printf("Enter number%d: ", i+1);
        scanf("%f", &nums[i]);
        sum += nums[i];
    }
    average = sum/n;
    printf("Sum = %.2f", sum);
    printf("Average = %.2f", average);
    return 0;
}
```

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,a[10],temp,j;

printf("Enter any 10 num in array = \n");
for(i=0;i<10;i++)
{ scanf("%d",&a[i]);
}
printf("\n\nData before sorting = ");
for(j=0;j<10;j++)
{
printf(" %d",a[j]);
}
for(i=0;i<10;i++)
{
for(j=0;j<10-i;j++)
{
if(a[j]>a[j+1])
{
temp=a[j];
a[j]=a[j+1];
a[j+1]=temp;
}
}
}
printf("\n\nData after sorting = ");
for(j=0;j<10;j++)
{
printf(" %d", a[j]);
}
getch();
}
```


Two Dimensional Arrays :

- កម្មវិធី C បានផ្តល់អោយមានការប្រើប្រាស់នូវ Array ពីរវិមាឌ (Two dimension arrays) ប៉ុន្តែវាត្រូវបានគេប្រើប្រាស់តិចជាង Array មួយវិមាឌ ។
- Syntax : **Data-Type Array_Name[Length 1] [Length 2];**
- Float dr[3] [5]; 3 rows 5 columns

dr[0,0]	dr[0,1]	dr[0,2]	dr[0,3]	dr[0,4]
dr[1,0]	dr[1,1]	dr[1,2]	dr[1,3]	dr[1,4]
dr[2,0]	dr[2,1]	dr[2,2]	dr[2,3]	dr[2,4]

```
#include<stdio.h>
int main()
{
    /* 2D array declaration*/ int abc[2][3];
    /*Counter variables for the loop*/ int i, j;
    for(i=0; i<2; i++)
    {
        for(j=0;j<3;j++)
        {
            printf("Enter value for abc[%d][%d]:", i, j);
            scanf("%d", &abc[i][j]);
        }
    }
    return 0;
}
```

នៅក្នុងការសរសេរកម្មវិធីធំៗ គឺយើងមិនអាចសរសេរដោយពុំបាន បំបែកកម្មវិធីនោះឲ្យទៅជាកម្មវិធីតូចៗទេ ព្រោះការមិនបំបែកកម្មវិធី នេះ នាំឲ្យមានការលំបាកនៅពេលកម្មវិធីមានភាព Error និងពិបាក ក្នុងការសរសេរ។ ការបំបែកកម្មវិធីឲ្យទៅជាតូចៗ នៅក្នុងកម្មវិធី C គេប្រើ Function

- Sub program
- វាមានលក្ខណៈងាយស្រួលក្នុងការគ្រប់គ្រង
- ងាយស្រួលកែនៅពេលមានភាព Error កើតឡើង
- កាត់បន្ថយទំហំ Memory
- ធ្វើឲ្យកម្មវិធីដំណើរការមានប្រសិទ្ធភាពខ្ពស់។

ប្រភេទ Function

នៅក្នុងភាសា C ឬ C++ Function ត្រូវបានចែកចេញជាពីរគឺ:

- Standard Library Function
- Function បង្កើតឡើងដោយ Programmer

១. **Standard Library Function:**

- ជាប្រភេទអនុគមន៍មួយដែលគេសរសេររួចជាស្រេច និង បញ្ចូលទៅក្នុងកុំព្យូទ័រនៃកម្មវិធី ។ ការហៅ Function ទាំងនេះមកប្រើ គឺទៅតាមរយៈ Header file ។ Standard Library Function គឺមិនអាចកែប្រែបានទេ។ <stdio.h>
- Standard Library ដែលយើងសិក្សាមានដូចជា៖ printf; scanf;...។

២. Function បង្កើតឡើងដោយ Programmer:

- ជាទូទៅ Function មានលក្ខណៈច្រើនប្រភេទ ដែលគេអាចស្គាល់វាបាន នៅពេលគេបង្កើតវា។ រាល់ Function ទាំងអស់ត្រូវមានពីរផ្នែកគឺ: Function's Header និង Function's Body។
- Function's Header មានបីផ្នែកគឺ:
- Data Type
- Function's Name
- List of Parameters in parentheses

Syntax

- Data_Type Fucntion_Name (Parameter's List)
- {
- Statement 1;
- Statement 2;
- .
- .
- Statement n;
- return expression;
- }

```
#include <stdio.h>
void functionName()
{
    ... ..
    ... ..
}
int main()
{
    ... ..
    ... ..
    .. ...
    functionName();
    ... ..
    ... ..
}
```


Function មិន Return value



Function ត្រឡប់ Return Value

```
void sum (int a, int b)
{
    printf ("Sum= %d",
a+b);
}
```

Example

- `#include<stdio.h>`
- `#include<conio.h>`
- `int sum (int, int);`
- `main()`
- `{`
- `int x, y, s;`
- `scanf ("%d %d", &x, &y);`
- `s = sum (x, y);`
- `printf ("sum = %d", s);`
- `return;`
- `}`
- `int sum (int a, int b)`
- `{`
- `return a + b;`
- `}`

```
#include <stdio.h>
void introduction()
{
    printf("Hi\n");
    printf("My name is
Chaitanya\n");
    printf("How are you?");

}
int main()
{
    /*calling function*/
    introduction();
return 0;
}
```

```
#include<stdio.h>
#include<conio.h>
void sum();
// declaring a function
clrscr();
int a=10,b=20, c;
void sum()
// defining function
{
    c=a+b;
    printf("Sum: %d", c);
}
void main()
{
    sum();
// calling function
}
```

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int table(int,int);
    int n,i;
    printf("Enter any num : ");
    scanf("%d",&n);
    for(i=1;i<=10;i++)
    {
        printf(" %d*%d= %d\n",n,i,table(n,i));
    }

    getch();
}
int table(n,i)
{
    int t;

    if(i==1)
    {
        return(n);
    }
    else
    {
        t=(table(n,i-1)+n);
        return(t);
    }
}
```

Local Variables

```
#include <stdio.h> int main () { /* local variable declaration */ int a, b; int c; /* actual initialization */ a = 10; b = 20; c = a + b; printf ("value of a = %d, b = %d and c = %d\n", a, b, c); return 0; }
```

- `#include <stdio.h>`
- `/* global variable declaration */`
- `int g;`
- `int main () {`
- `/* local variable declaration */`
- `int a, b;`
- `/* actual initialization */`
- `a = 10;`
- `b = 20;`
- `g = a + b;`
- `printf ("value of a = %d, b = %d and g = %d\n", a, b, g);`
- `return 0;`
- `}`

- `#include <stdio.h>`
- `/* global variable declaration */`
- `int g = 20;`
-
- `int main () {`
- `/* local variable declaration */`
- `int g = 10;`
-
- `printf ("value of g = %d\n", g);`
-
- `return 0;`
- `}`

- `#include <stdio.h>`
- `/* global variable declaration */`
- `int a = 20;`
- `int main () {`
- `/* local variable declaration in main function */`
- `int a = 10;`
- `int b = 20;`
- `int c = 0;`
- `printf ("value of a in main() = %d\n", a);`
- `c = sum(a, b);`
- `printf ("value of c in main() = %d\n", c);`
- `return 0;`
- `}`
- `/* function to add two integers */`
- `int sum(int a, int b) {`
- `printf ("value of a in sum() = %d\n", a);`
- `printf ("value of b in sum() = %d\n", b);`
- `return a + b;`
- `}`

```
#include<stdio.h>
```

```
/*function declarations*/
```

```
int    sumTwoNum(int,int); /*to get sum*/
```

```
float  averageTwoNum(int,int); /*to get average*/
```

```
int main()
```

```
{
```

```
    int number1,number2;
```

```
    int sum;
```

```
    float avg;
```

```
    printf("Enter the first integer number: ");
```

```
    scanf("%d",&number1);
```

```
    printf("Enter the second integer number: ");
```

```
    scanf("%d",&number2);
```

```
    /*function calling*/
```

```
    sum=sumTwoNum(number1,number2);
```

```
    avg=averageTwoNum(number1,number2);
```

```
    printf("Number1: %d, Number2: %d\n",number1,number2);
```

```
    printf("Sum: %d, Average: %f\n",sum,avg);
```

```
    return 0;
```

```
}
```

```
int sumTwoNum(int x,int y)
```

```
{
```

```
    /*x and y are the formal parameters*/
```

```
    int sum;
```

```
    sum=x+y;
```

```
    return sum;
```

```
}
```

```
float averageTwoNum(int x,int y)
```

```
{
```

```
    /*x and y are the formal parameters*/
```

```
    float average;
```

```
    return ((float)(x)+(float)(y))/2;
```

```
}
```

```

#include<stdio.h>

#define MAX_ELEMENTS 100

/*function declaration*/

int sumOfElements(int [],int);

int main()
{
    int N,i,sum;
    int arr[MAX_ELEMENTS];

    printf("Enter total number of elements(1 to %d): ",MAX_ELEMENTS);
    scanf("%d",&N);

    if(N>MAX_ELEMENTS)
    {
        printf("You can't input larger than MAXIMUM value\n");
        return 0;
    }
    else if(N<0)
    {
        printf("You can't input NEGATIVE or ZERO value.\n");
        return 0;
    }

    /*Input array elements*/
    printf("Enter array elements:\n");
    for(i=0; i<N; i++)
    {
        printf("Enter element %4d: ",(i+1));
        scanf("%d",&arr[i]);
    }

    /*function calling*/
    sum=sumOfElements(arr,N);

    printf("\nSUM of all Elements: %d\n",sum);

    return 0;
}

```

```

int sumOfElements(int x[],int n)
{
    int sum,i;
    sum=0;

    for(i=0; i<n; i++)
    {
        sum += x[i];
    }

    return sum;
}

```

```
#include<stdio.h>
```

```
/*function declaration*/  
int stringLength(char *);
```

```
int main()  
{
```

```
    char text[100];  
    int length;
```

```
    printf("Enter text (max- 100 characters): ");  
    scanf("%^[^\\n]s",text);  
    /*we can also use gets(text) - To read complete text untill '\\n'*/
```

```
    length=stringLength(text);
```

```
    printf("Input text is: %s\\n",text);  
    printf("Length is: %d\\n",length);
```

```
    return 0;
```

```
}
```

```
int stringLength(char  
*str)
```

```
{
```

```
    int len=0;
```

```
    /*calculate string  
length*/
```

```
    for(len=0;  
str[len]!='\\0'; len++);
```

```
    /*return len*/
```

```
    return len;
```

```
}
```

- `#include<stdio.h>`
- `void funA();`
- `void funB();`
- `void funC();`
- `void funD();`
- `int main()`
- `{`
- `printf("Hey from main function\n");`
- `funA();`
- `printf("Bye from main function\n");`
- `}`
- `void funA()`
- `{`
- `printf("Hey from main function A\n");`
- `funB();`
- `printf("Bye from main function A\n");`
- `}`

- void funB()
- {
- printf("Hey from main function B\n");
- funC();
- printf("Bye from main function B\n");
- }
- void funC()
- {
- printf("Hey from main function C\n");
- funD();
- printf("Bye from main function C\n");
- }
- void funD()
- {
- printf("Hey from main function D\n");
- printf("Bye from main function D\n");
- }

- #include<stdio.h>
- #include<conio.h>
- int main(){
- char name [50];
- int Salary,Sell;
- float RecieveMoney, Commission;
- printf("Input Name= ");
- scanf("%s", &name);
- printf("Input Salary= ");
- scanf("%d",&Salary);
- printf("Input Sell= ");
- scanf("%f",&Sell);
- if(Sell>=10000){
- Commission= 0.05 * (Salary)+100;
- printf("\nCommission=%f",Commission);
- }
- if(Sell<10000){
- printf("\nNo Commission");
- }RecieveMoney=Salary+Commission;
- printf("\nRecieveMoney=%f", RecieveMoney);
- getch();

- `#include<stdio.h>(Do Wile Loop)`
- `#include<conio.h>`
- `int main()`
- `{`
- `int a,b,pgcd;`
- `printf("\Input a,b:");`
- `scanf("%d%d",&a,&b);`
- `do`
- `{`
- `if(a>b)`
- `a=a-b;`
- `else`
- `b=b-a;`
- `} while(a!=b);`
- `pgcd=a;`
- `printf("\nThe PGCD of a and b is%d",pgcd);`
- `getch();`
- `}`

- `#include<stdio.h>`
- `#include<conio.h>`
- `int main()`
- `{`
- `int salary,Tax;`
- `printf("Input salary:");`
- `scanf("%d",salary);`
- `if(salary>3000)`
- `Tax=250+0.015 * (salary-3000);`
- `else if(salary>2000)`
- `Tax=150+0.01 * (salary-2000);`
- `else`
- `Tax=150;`
- `printf("\nTax=%d",Tax);`
- `getch();`
- `}`

- `#include<stdio.h>`
- `#include<conio.h>`
- `int main()`
- `{`
- `int st,et,Amount;`
- `zprintf("Enput st,et:");`
- `scanf("%d%d",&st,&et);`
- `if(st<16)`
- `if(et<16)`
- `Amount=(et-st)*100;`
- `else`
- `Amount=(16-st)*100+(et-16)*200;`
- `else`
- `Amount=(et-st)*200;`
- `printf("The Amount is %d",Amount);`
- `getch();`
- `}`