

C PROGRAMMING

រៀបរៀងដោយ



លោក លី ឆែណាក់



Tel : 070 908 739

Email : Kompongsom4u@gmail.com

Website : www.kompongsom4u.blogspot.com



មេរៀនទី ១ ៖ ការណែនាំស្តាប់ពី C PROGRAMMING

១.១. និយមន័យ

C PROGRAMMING គឺជា Program Language មួយដែលប្រើប្រាស់សំរាប់សរសេរក្នុងការបង្កើតនូវសំនុំពាក្យមួយចំនួន ដើម្បីធ្វើការដាក់លាក់ណាមួយ ។

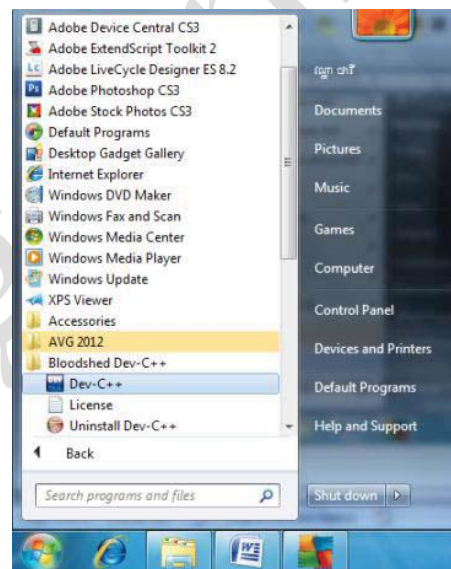
១.២. ការប្រើប្រាស់កម្មវិធី Dev C++

ខាងក្រោមនេះជារបៀបក្នុងការបើកកម្មវិធី Dev C++ ដូចនេះសូមអនុវត្តន៍ដូចខាងក្រោម

១. ចុច Start Menu

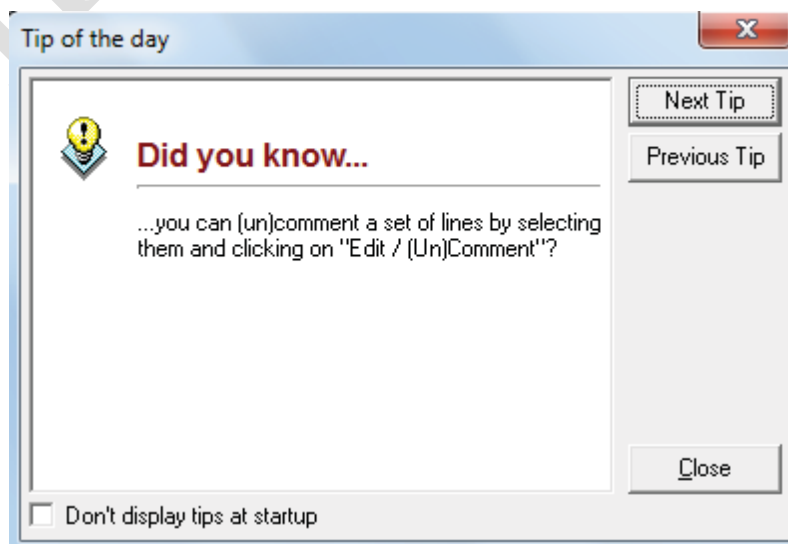
២. All Program

៣. Bloodshed Dev C++



៤. សូម Tick ក្នុងប្រអប់ Don't Display tips as Startup

៥. ចុច Close



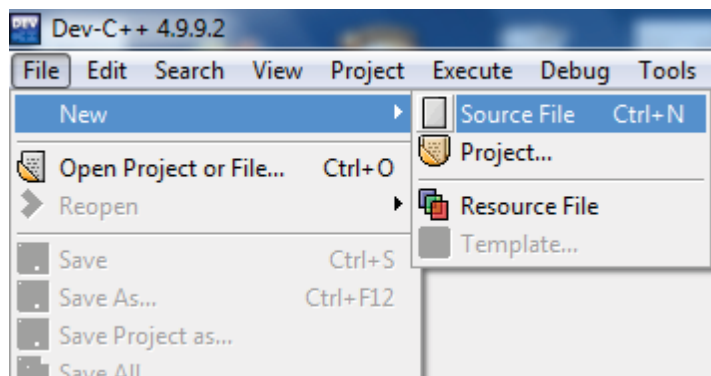
១.៣. របៀបបង្កើត Source File

ដើម្បីបង្កើត Source File សូមអនុវត្តដូចខាងក្រោម ៖

១. ចុច File Man u

២. New

៣. Source File

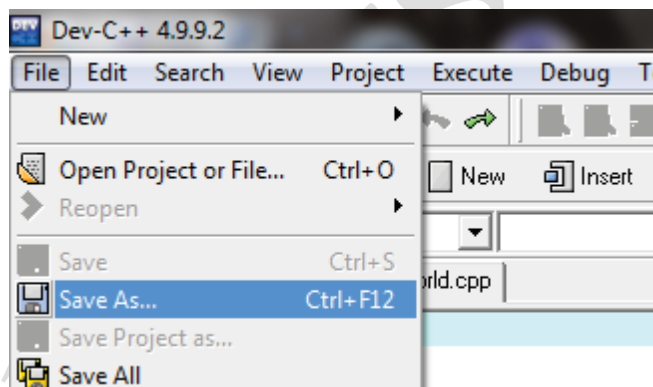


១.៤. របៀបរក្សា Source File

ដើម្បីរក្សា Source File សូមអនុវត្តដូចខាងក្រោម ៖

១. ចុច File Man u

២. Save ឬ Save As



៣. ត្រង់ File Name សូមដាក់ឈ្មោះ (nak.cpp)

៤. ជ្រើសរើសទីតាំងក្នុងការដាក់ File

៥. ចុច Button Save

១.៤. ការសរសេរ Cord ដំបូងគេ

Source Code:

```
#include<stdio.h>
#include<conio.h>
main() {
    printf ("Hello chhynak.\n") ;
    printf ("How Are You.\n") ;
    getch() ;
    return (0) ;
}
```

Output

```
Hello chhaynak
How are you
```



១.៥. កន្លែងអត្ថន័យនៃ Source Code

- **#include<stdio.h>** : វាគឺជា Standard Input/output ដែលផ្ទុកនូវបណ្តា Function ដូចជា Printf មានន័យអាចប្រើ Printf បានលុះត្រាតែមាន #include<stdio.h> នេះ ឬក៏គេហៅថាជា Header file ក៏បានដែរ ។

- **#include<conio.h>** : វាគឺជា Functionក្នុងការផ្ទុក Getch(); អាចប្រើ Printf បានលុះត្រាតែមាន #include<stdio.h>

- **main(){} :** វាគឺជា Main Function ជាចំនុចដំណើរការ Code នៅក្នុង Program ទាំងអស់ ។

- **Printf :** ជាការបង្ហាញទិន្នន័យចេញមកក្រៅលើ Command CMD

- **Getch() :** សំរាប់បញ្ឈប់ដំណើរការ Code ពេលដែល Press Anykey វានឹងបន្តដំណើរការទៀត

១.៦. រូបមន្តសរសេរ CORD

```
#include<stdio.h>
#include<conio.h>
main() {
    Statement;
    .....
    getch();
    return (0);
}
```

Ex.

```
#include<stdio.h>
#include<conio.h>
main() {
    printf ("Hello chhynak.\n") ;
    printf ("How Are You.\n") ;
    getch();
    return (0);
}
```

១.៧. របៀបប្រើ Character

Escape character ឬ Escape code គឺជាតួអក្សរដែលប្រើប្រាស់ក្នុង source code ។ ដូចជាការ ចុះបន្ទាត់ ដកឃ្លា ជាដើម ។

Code	អត្ថន័យនៃ Code
\t	tab ចូលបន្ទាត់
\a	alert (beep)
\n	Newline ចុះបន្ទាត់
\'	single quote (')
\?	question mark (?)



ខ.

```
#include <stdio.h>
#include<conio.h>
main(){
printf("Hello ! this is:\t \"chhaynak\"\\n");
printf("What\'s are you doing?\\n");
printf("Working");
getch();
return (0);
}
```

Output

```
Hello ! this is chhaynak
what's are you doing?
Working
```

១.៨. របៀបសរសេរជា COMMENT

ចំពោះការប្រើប្រាស់ Command ក៏ជាចំនុចមួយប្រសើរដែរ ចំពោះ Programmer ព្រោះថា Command អាចប្រាប់ពីគោលបំណងរបស់ Code នឹង ថា វា សរសេរបែបនេះដើម្បីធ្វើអ្វី

```
/* this command for tell
you what is this > */
```

១.៩. លំហាត់សាកល្បង

ចូលសរសេរកម្មវិធីតូចមួយអោយមានកាងបង្ហាញទិន្នន័យដូចខាងក្រោមនេះ ៖

១. ឈ្មោះបុគ្គលិក

២. ភេទ

៣. សញ្ញាតិ

៤. ទីកន្លែងកំណើត

៥. អាសយដ្ឋាន

៦. ប្រាក់ខែ

៧. លេខទូរស័ព្ទ

ចប់មេរៀនទី ១



មេរៀនទី ២ ៖ អថេរ និងប្រភេទទិន្នន័យ

២.១ ប្រភេទទិន្នន័យ

ប្រភេទទិន្នន័យត្រូវបានប្រើប្រាស់ ដើម្បីកមណត់តម្លៃរបស់អថេរ ។ គេច្រើនប្រើនៅពេលប្រកាសអថេរ ។
សូមមើលរូបមន្តដូចខាងក្រោម ៖

Syntax : Data type Variable

Ex. int a;

ខាងក្រោមនេះជាប្រភេទទិន្នន័យ ៖

Name	Description	Size	Range
Short	Short integer	2byte	Signed:-32768 to 32767
Int	integer	4byte	Signed:2147483648 to 214748347
Long	Long integer	4byte	Signed:2147483648 to 214748347
Float	Float point number	4 byte	-3.4×10^{-38} TO 3.4×10^{-38}
Double	Double precision floating point number	8 byte	$\pm 1.7e \pm 308$ (~15digits)
Boolean	Boolean value	1 byte	True or False
Char	Character or small integer	1byte	Signed:-128 to 127

Code ខាងក្រោមនេះសំរាប់ Test មើលទំហំរបស់ Data type

```
#include<stdio.h>
#include<conio.h>
main() {
    printf( "Fundamental of size of operator.\n\n" );
    printf( "Size of char %3d byte.\n", sizeof( char ) );
    printf( "Size of int %3d byte.\n", sizeof( int ) );
    printf( "Size of long %3d byte.\n", sizeof( long ) );
    printf( "Size of short %3d byte.\n", sizeof( short ) );
    printf( "Size of signed %3d byte.\n", sizeof( signed ) );
    printf( "Size of unsigned %3d byte.\n", sizeof( unsigned ) );
    printf( "Size of float %3d byte.\n", sizeof( float ) );
    printf( "Size of double %3d byte.\n", sizeof( double ) );
    printf( "Size of long double %3d byte.\n", sizeof( long double ) );
    printf( "\nPress < Enter to Exit ! >" );
    getch( );
    return ( 0 );
}
```



២.២. ឈ្មោះអថេរ

អថេរគឺជាឈ្មោះដែលតំណាងអោយតំបន់ដែលផ្ទុកទិន្នន័យ (Data) ក្នុងអង្គចងចាំ របស់កុំព្យូទ័រ ។ ការប្រកាសអថេរត្រូវមានលក្ខណៈដូចខាងក្រោម ៖

- អាចប្រើតួពី A ទៅ Z (តូច ទៅ ធំ) លេខ និង Underscore
- ត្រូវអក្សរ ត្រូវតែជាតួអក្សរ ឬ Underscore
- អក្សរតូច ឬ ធំ នៅក្នុង Variable Name ពី ដែលមានឈ្មោះដូចគ្នា ជាអថេរផ្សេងគ្នា
- មិនត្រូវមានលក្ខណៈដូច Keyword ។ Keyword មានដូចជា ៖

Array for return break function sizeof
case goto if constants switch continue
labels pointer union

សូមពិនិត្យមើលឧទាហរណ៍ ដែលការប្រកាសអថេរមិនត្រឹមត្រូវ ៖

- total\$: (Illegal character \$) មិនអាចប្រើសញ្ញា \$
- 2nd_sums : (Begin with a digit) មិនអាចចាប់ផ្តើមដោយលេខមុន
- second sum : (Cannot use blank as a character) មិនអាចដកហូត
- Total-Cars : (Illegal character -) មិនអាចប្រើប្រាស់សញ្ញា “-”

ដើម្បីធ្វើការ Output នូវតម្លៃរបស់អថេរ នៅក្នុង Printf(); Fucntion យើងប្រើ Formation Control Data Type ដែលមានបង្ហាញនៅក្នុងតារាងខាងក្រោម ៖

Quantity	Data Type
%d, %i	Integer (int) ចំនួនគត់
%c	Single character (char) តួអក្សរមួយតួ
%s	Character string (char) តួអក្សរប្រើនតួ
%f	Fractional number (float) ចំនួនទស្សភាគ
%lf	Double fractional number (double) ចំនួនទស្សភាគ
%ld	Long Integer (long) ចំនួនគត់

២.៣. ការប្រកាសអថេរ

គ្រប់អថេរទាំងអស់ត្រូវតែ ប្រកាសជាមុនសិន មុនពេលយកទៅប្រើក្នុងការសរសេរកម្មវិធី ។ ការប្រកាសអថេរ ផ្ដោតសំខាន់ទៅលើប្រភេទទិន្នន័យ និងចំនួនអថេរមានមួយចំនួន ។ សូមមើលការប្រកាសខាងក្រោមនេះ ៖

Ex. int a,b,sum;
 float c;

ការប្រកាសអថេរមួយ ដែលផ្តល់តម្លៃអោយអថេរមួយទៀត សូមមើលឧទាហរណ៍ខាងក្រោមនេះ ៖

Ex char c = 'A';
 int i = 0;
 int k = i + 1;



២.៣. ការធ្វើប្រមាណវិធី

1. Arithmetic Operator: គឺជា Operator សមញ្ញមួយ និងជំនួយដែលគេតែងតែប្រើធ្វើជាប្រមាណវិធីក្នុងការសរសេរ Program ។ Arithmetic Operator មាន៖

Operator	Meaning
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus [remainder of an integer division.]

Arithmetic Operator មានដូចជា ៖ បូក ដក គុណ ចែក គឺគេប្រើទៅនឹងដូចការប្រើប្រាស់ធម្មតាដែរ។ ដោយឡែកប្រមាណវិធី Modulus គឺជាប្រមាណវិធីដែលទទួលតម្លៃពីសំណល់ចែក ។

```
ឧទាហរណ៍    int a=1;
               int b=6;
               int c= a+b;
```

2 – Relational and Equality Operator: គឺជាប្រមាណវិធីសំរាប់ធ្វើការប្រៀបធៀបនៃពីចំនួនដូចគ្នា។ គេប្រើប្រើប្រាស់វា ដើម្បីពិនិត្យលក្ខខណ្ឌនៅក្នុង Loop Statement , if-else Statement ,.....។

Relational and Equality Operator មានដូចជា ៖

< តូចជាង (Less than) <= តូចជាងឬស្មើ (Less than or Equal)
 > ធំជាង (Greater than) >= ធំជាងឬស្មើ (Greater than or Equal)
 == ស្មើ !=មិនស្មើ

3 – Logical Operator: គឺជាប្រមាណ ដែលគេច្រើនប្រើភ្ជាប់ខ័ណ្ឌពីរផ្សេងគ្នា

Logical Operator មានដូចជា ៖

&& ឈ្មាប់នឹង
 || ឈ្មាប់ឬ

4. Increment and Decrement Operator : កម្មវិធី C បានផ្តល់អោយមានប្រមាណវិធីផ្សេងទៀតដែលប្រើសំរាប់ធ្វើការកំនើន (Increment) និង បន្ថយ (Decrementing) តម្លៃរបស់អថេរ ។ ប្រមាណវិធី Increment (++) និងបូកបន្ថែមមួយ នៅពេលដែលធ្វើការគណនា ។ ប្រមាណវិធី Decrement (--) និងបូកបន្ថែមមួយ នៅពេលដែលធ្វើការគណនា ។

ប្រមាណវិធីទាំងពីរនេះត្រូវបានប្រើប្រាស់តែមួយអថេរប៉ុណ្ណោះ ។ គេមិនប្រើជាមួយ Constant ឬកន្សោមទេ ។ ចំពោះប្រមាណវិធីទាំងពីរនេះ យើងអាចប្រើនៅពីមុខអថេរ (++n)ក៏បាន ឬពីក្រោយអថេរក៏បាន (n++)។

បញ្ជាក់៖ -(++n) មានន័យថាកើនមុនពេលប្រើប្រាស់ (Ex. n=5 , x=++n , ពេលនោះ x=6)

-(n++)មានន័យថាកើនក្រោយពេលយកប្រើ (Ex. x=n++, x=5, n=6)

ឧទាហរណ៍២ :




```
#include<stdio.h>
main ( ) {
    int a = 5;
    int b = 5;
    printf( "Value of a: %d and b: %d\n", a, b );
    printf( "Value of a: %d\n", ++a );
    printf( "Value of b: %d\n", b++ );
    getch();
    return ( 0 );
}
```

ប្រមាណវិធី Prefix និង Postfix នឹងដូចគ្នានៅពេលដែលយើងប្រើវាតែឯង

5. Assignment operator: គេមានប្រមាណវិធីមួយចំនួនទៀតនៅលើប្រមាណវិធីធម្មតា ដែលគេប្រើសំរាប់ Assigned

(=) [ផ្ទេរតំលៃ] ឬគណនាកន្សោមផ្សេងៗ ។ ប្រមាណវិធីនេះគឺ :

Operator	Simple	Equal	Meaning
+=	a+=b	a = a + b	a បូកបន្ថែម b ហើយ a ទទួលតម្លៃថ្មី
-=	a-=b	a = a - b	a ដកបន្ថែម b ហើយ a ទទួលតម្លៃថ្មី
=	a=b	a = a * b	a គុណបន្ថែម b ហើយ a ទទួលតម្លៃថ្មី
/=	a/=b	a = a / b	a ចែកបន្ថែម b ហើយ a ទទួលតម្លៃថ្មី
%=	a=b%b	a = a % b	a ចែកអោយ b ហើយ a ទទួលយកសំណល់ពីការចែករវាង a និង b

ឧទាហរណ៍៣ :

```
#include<stdio.h>
main ( ) {
    int a = 1;
    int b = 5;
    printf( "Value of a: %d and b: %d\n", a, b );
    a += b;
    printf( "Value of a: %d\n", a );
    b %= 3;
    printf( "Value of b: %d\n", b );
    b *= a + 1;
    printf( "Value of b: %d\n", b );
    b /= a - 1;
    printf( "Value of b: %d\n", b );
    getch();
    return ( 0 );
}
```

លំហាត់ ៖ ចូលបង្ហាញពី Output នៃ Program ខាងក្រោម ៖

1. ចូលសរសេរកម្មវិធីដើម្បីគណនា ៖

-ក្រឡាផ្ទៃចតុកោណកែង (S=a*b)

-គណនាផលបូក S=a+b+d

-គណនាក្រឡាផ្ទៃក្រឡា S=a*a



មេរៀនទី ៣ ៖ Control Structure

៣.១.If Statement

បើសិន Expression ពិត (Expression មានតម្លៃមិនស្មើសូន្យ) Statement ត្រូវបានអនុវត្តន៍ តែផ្ទុយទៅវិញ ទៅវិញ បើ Expression មិនពិត (Expression មានតម្លៃស្មើសូន្យ) វានឹងអនុវត្តន៍ Statement 2 ។ នៅក្នុងភាសា C Programming Statement មួយត្រូវបានបញ្ចប់ដោយសញ្ញា Semicolon “;” តែបើជាមួយ Block Statement វិញគេ ត្រូវប្រើសញ្ញា Braces “{និង}” ដើម្បីបិទនិងបើក ។ ដូច្នេះ if Statement មាន Syntax ដូចខាងក្រោម ៖

Syntax1:

```
if(expression ) Statement;
Statement2;
```

Syntax2:

```
if(expression ) {
    Statement1;
    Statement2;
    .....
    .....
}
```

ឧទាហរណ៍ ១ ៖

```
#include<stdio.h>
main( ) {
    int x;
    clrscr( );
    printf( "Input x:" );
    scanf( "%d", &x );
    if ( x == 10 ) {
        printf( "\n This is a good student." );
    }
    return ( 0 );
}
```

ឧទាហរណ៍ ២ ៖

```
#include<stdio.h>
#include<stdlib>
main( ) {
    int magic, guess;
    printf( "\nPlease input Guess value: " );
    scanf( "%d", &guess );
    magic = rand( );
    if ( guess == magic )
        printf( "\n You are right." );
    if ( guess < magic )
        printf( "\n You are too low." );
    if ( guess > magic )
        printf( "\n You are too high." );
    printf( "\n %d is the magic value", magic );
    getch( );
    return 0;
}
```



៣.២.If –else Statement

បើសិនជា Expression ពិត Statement 1 ត្រូវបានអនុវត្តន៍ តែបើ Statement មិនពិតវិញ Statement2 អនុវត្តន៍វិញ ។

រូបមន្ត :

```
if(expression ) {
    Statement1;
}
else{
    Statement2;
}
```

ឧទាហរណ៍ ១ ៖

```
វក្កពន្ធកសិស្ស
#include<stdio.h>
#include<conio.h>
main() {
    if (result >= 45){
        printf("Pass\n");
    }
    else{
        printf("Fail\n");
        getch();
    }
}
```

ឧទាហរណ៍ ២ ៖

```
វក្កសមីការឌីក្រេទី ២
#include<conio.h>
#include<math.h>
void main() {
    clrscr();
    int a,b,c;
    float Deta,x1,x2;
    printf("Input A=");
    scanf("%d",&a);
    printf("Input B=");
    scanf("%d",&b);
    printf("Input C=");
    scanf("%d",&c);
    Deta=(pow(b,2)-(4*a*c));
    if(Deta==0){
        x1=x2=(-b/(2*a));
        printf("x1=x2=%0.2f\n",x1);
    }
    else if(Deta<0)
        printf("No root\n");
    else{
        x1=(-b-sqrt(Deta))/(2*a);
        x2=(-b+sqrt(Deta))/(2*a);
        printf("X1 is=%0.1f\n",x1);
        printf("X2 is=%0.1f\n",x2);
    }
    getch();
}
```



ឧទាហរណ៍ ៣ ៖ ចូលអ្នកសរសេរកម្មវិធីមួយគិតលុយការទិញសំភារៈដូចខាងក្រោម៖

- 1...Book** [-បើទិញសៀវភៅក្រោម ១៥ ក្បាល មិនចុះតម្លៃ
 -បើទិញសៀវភៅលើសពី ១៥ ដល់ ៣០ ក្បាល ចុះតម្លៃ ៥ %
 -បើទិញសៀវភៅលើសពី ៣០ ដល់ ៦០ ក្បាល ចុះតម្លៃ ១០ %]
- 2...Pen** [-បើទិញប៊ិចក្រោម ១៥ ដើម មិនចុះតម្លៃ
 -បើទិញប៊ិចលើសពី ១៥ ដល់ ៣០ ដើម ចុះតម្លៃ ៥ %
 -បើទិញប៊ិចលើសពី ៣០ ដល់ ៦០ ដើម ចុះតម្លៃ ១០ %]
- 2...Ruler** [-បើទិញបន្ទាត់ក្រោម ១៥ ដើម មិនចុះតម្លៃ
 -បើទិញបន្ទាត់លើសពី ១៥ ដល់ ៣០ ដើម ចុះតម្លៃ ៥ %
 -បើទិញបន្ទាត់លើសពី ៣០ ដល់ ៦០ ដើម ចុះតម្លៃ ១០ %]

```
#include<stdio.h>
#include<conio.h>
#include<process.h>
void main() {
    Again:
    clrscr();
    int book,pen,ruler,U_price;
    float t_book,t_pen,t_ruler,t_all;
    char ch;

    gotoxy(33,2);
    textbackground(GREEN);
    textcolor(RED);
    cprintf(" THINKING MONEY ");

    //BOOK

    gotoxy(27,4);
    printf("Input Qauntity Book: ");
    scanf("%d",&book);

    gotoxy(27,5);
    printf("Input unit price : ");
    scanf("%d",&U_price);

    if(book<=15){
        t_book=book*U_price;
        gotoxy(25,6);
        textcolor(YELLOW);
        cprintf("You have to pay = %0.1f Rile\n",t_book);
    }
    if(book>15 && book<=30){
        t_book=(book*95/100)*U_price;
        gotoxy(25,6);
        textcolor(YELLOW);
        cprintf("You have to pay = %0.1f Rile\n",t_book);
    }
    if(book>30 && book<=60){
        t_book=(book*90/100)*U_price;
        gotoxy(25,6);
        textcolor(YELLOW);
        cprintf("You have to pay = %0.1f Rile\n",t_book);
    }
}
```



```

//PEN
gotoxy(27,8);
printf("Input Qauntity Pen      : ");
scanf("%d",&pen);
gotoxy(27,9);
printf("Input unit price        : ");
scanf("%d",&U_price);
if(pen<=15){
    t_pen=pen*U_price;
    gotoxy(25,10);
    textcolor(YELLOW);
    cprintf("You have to pay =%0.1f Rile\n",t_pen);
}
if(pen>15 && pen<=30){
    t_pen=(pen*95/100)*U_price;
    textcolor(YELLOW);
    gotoxy(25,10);
    cprintf("You have to pay = %0.1f Rile\n",t_pen);
}
if(pen>30 && pen<=60){
    t_pen=(pen*90/100)*U_price;
    textcolor(YELLOW);
    gotoxy(25,10);
    cprintf("You have to pay = %0.1f Rile\n",t_pen);
}

//Ruler
gotoxy(27,16);
printf("Input Qauntity Ruler   : ");
scanf("%d",&ruler);
gotoxy(27,17);
printf("Input unit price        : ");
scanf("%d",&U_price);
if(ruler<=15){
    t_ruler=ruler*U_price;
    textcolor(YELLOW);
    gotoxy(25,18);
    cprintf("You have to pay = %0.1f Rile\n",t_ruler);
}
if(ruler>15 && ruler<=30){
    t_ruler=(ruler*95/100)*U_price;
    textcolor(YELLOW);
    gotoxy(25,18);
    cprintf("You have to pay = %0.1f Rile\n",t_ruler);
}
if(ruler>30 && ruler<=60){
    t_ruler=(ruler*90/100)*U_price;
    textcolor(YELLOW);
    gotoxy(25,18);
    cprintf("You have to pay = %0.1f Rile\n",t_ruler);
}
t_all=t_book+t_pen+t_ruler;
textcolor(YELLOW);
gotoxy(25,20);
cprintf("TOTAL All TO PAY = %0.1f Rile",t_all);

gotoxy(31,24);
textcolor(YELLOW+GREEN);
cprintf("Continue Program (Y/N)?");
fflush(stdin);
ch=getchar();
if(ch=='Y' || ch=='y')
    goto Again;
if(ch=='N' || ch=='n')
    exit(1);

}

```

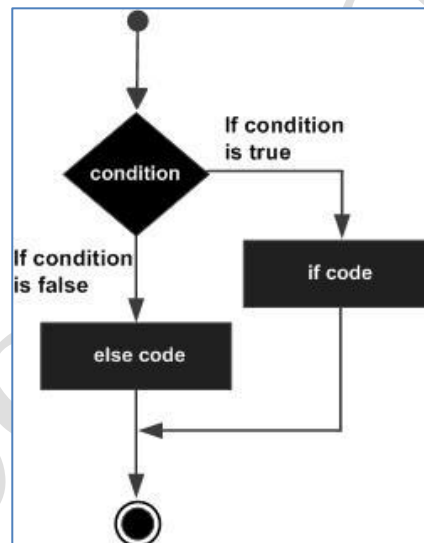


៣.៣.If –else –if Statement

if – else_if Statement : ត្រូវបានប្រើប្រាស់ដូច if-else statement គេត្រូវបានប្រើប្រាស់សំរាប់ដាក់លក្ខខណ្ឌច្រើន (លើសពី ២) ។ ទំរង់ទូទៅរបស់នៃ if-else-if Statement ដូចខាងក្រោម ៖

រូបមន្ត :

```
if ( expression 1)
    statement 1;
else if ( expression 2)
    statement 2;
else if ( expression 3)
    statement 3;
else if ( expression n-1)
    statement n-1;
else
    statement n;
}
```



ឧទាហរណ៍ ១ ៖ បង្ហាញ Message នៅក្នុង ឬ ចាស់ អាស្រ័យការបញ្ចូល លេខ

```
#include <iostream>
#include <conio.h>
int main(){
    int age;
    scanf("%d",&age);
    if ( age < 100 ) {
        printf("You are pretty young!\n");
    }
    else if ( age == 100 ) {
        printf("You are old\n");
    }
    else {
        printf("You are really old\n");
    }
}
```

ឧទាហរណ៍ ២ ៖ ការប្រៀបធៀបតម្លៃឯក

```
#include<stdio.h>
int main(){
    int x,y;
    printf("Enter value for x :");
    scanf("%d",&x);
    printf("Enter value for y :");
    scanf("%d",&y);
    if ( x > y ){
        printf("X is large number - %d\n",x);
    }
    else{
        printf("Y is large number - %d\n",y);
    }
    return 0;
}
```

ឧទាហរណ៍ ៣ ៖ ការរកភាពជាប់ធ្លាប់សិស្ស

```
#include
main()
{
    int grade;
    char student[25],section[25];

    clrscr();
    printf("Enter the name of a student: ");
    scanf("%s", &student);
    printf("Enter the section of a student: ");
    scanf("%s", &section);
    printf("Enter the grade of a student: ");
    scanf("%d", &grade);

    if(grade>75)
        printf("PASSED");
    else
        print("FAILED");

    getch();
}
```



៣.៤ .Switch Statement

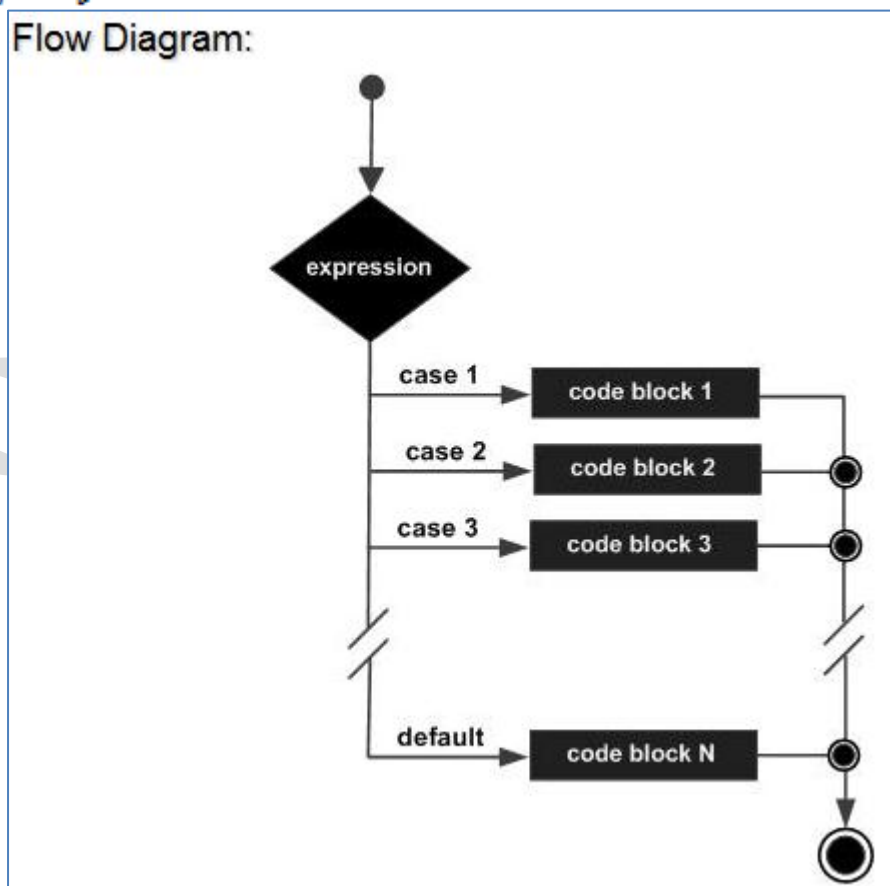
Switch Statement ត្រូវបានគេប្រើប្រាស់ដូច if-else-if ដែរ គឺថាពេលដែល if-else-if Statement មានការត្រួតពិនិត្យលើ Variable តែមួយគត់ លើលក្ខខណ្ឌ ពេលនោះយើងប្រើ Switch Statement មកជំនួសវិញ ។ រូបមន្ត

```
switch ( Variable ) {
    case constant 1:
        Statement(s);
        break;
    case constant 2:
        Statement(s);
        break;
    .
    case constant n:
        Statement(s);
        break;
    default:
        statement(s);
        break;
}
```

switch Statement បានយកតំលៃរបស់ Variable មកប្រៀបធៀបជាមួយតំលៃរបស់ Case

(Constant របស់ Case) គឺមួយៗ ថាគេតំលៃមួយណាដែលស្មើជាមួយតំលៃរបស់ Variable ហើយវាទាំង Execute ទៅលើ block statement របស់ Case នោះ។ តែបើតំលៃរបស់ Variable ពុំស្មើនឹងតំលៃរបស់ Case ណាមួយទេ វាទាំង Execute ទៅលើ Block statement របស់ default។ រាល់ Block statement ទាំងអស់របស់ Case និង default ត្រូវតែបញ្ចប់ដោយ break Statement ។

Flow Diagram:



Ex 1.

```
#include <stdio.h>

int main ()
{
    /* local variable definition */
    char grade = 'B';

    switch(grade)
    {
        case 'A' :
            printf("Excellent!\n" );
            break;
        case 'B' :
        case 'C' :
            printf("Well done\n" );
            break;
        case 'D' :
            printf("You passed\n" );
            break;
        case 'F' :
            printf("Better try again\n" );
            break;
        default :
            printf("Invalid grade\n" );
    }
    printf("Your grade is  %c\n", grade );

    return 0;
}
```

Ex 2.

```
#include<stdio.h>
#include<conio.h>
void main(){
    clrscr();
    int n;
    printf("Input N=");
    scanf("%d",&n);
    switch(n){
        case 1:
            printf("Number one");
            break;
        case 2:
            printf("Number two");
            break;
        case 3:
            printf("Number three");
            break;
        case 4:
            printf("Number four");
            break;
        default:
            printf("The Number biggest is=%d");
            }
        getch();
    }
```



៣.៥ .លំហាត់

១. ចូលសរសេរ Output Code ខាងក្រោម ៖

```
int x,y;
x=6;
y=1;
if(x<2)
    printf("%d\n",x);
else
    printf("%d\n",y);
```

២. ចូរឆ្លើយលំហាត់ខាងក្រោម

- សរសេរកម្មវិធីគណនាសមីការ $Ax+b=0$
- សរសេរកម្មវិធីដោយប្រើភាសា C ដើម្បីរកមធ្យមភាគគឺរវាងពិន្ទុកមុខវិជ្ជា ភាសាខ្មែរ និងគណិត
- សរសេរកម្មវិធីដោយប្រើ Switch ដាក់តម្លៃលេខអោយចេញថ្ងៃ

ចប់មេរៀនទី ៣



មេរៀនទី ៤ ៖ Loop Control Structure



នៅក្នុងផ្នែកនេះ យើងនឹងបង្ហាញពីដំណើរការ Repeat Statement ឬ មួយ Block Statement ដោយប្រើលក្ខណៈអ្វីមួយ ហៅថា Loop ។ ក្នុងភាសា C Loop មានបី ប្រភេទ គឺ ៖

១. While Loop Statement

២. do_while Loop Statement

៣. For Loop Statement

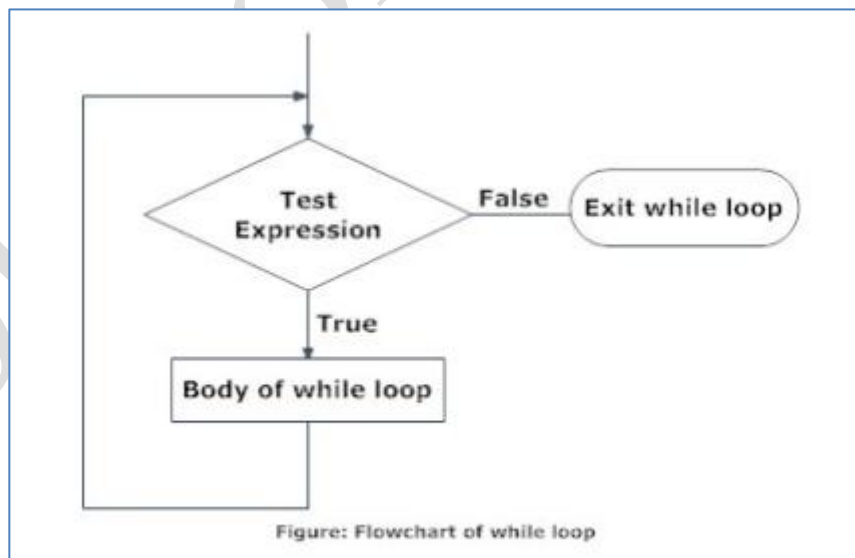
Loop គឺជាក្រុមនៃ Instruction Computer មួយដែល សកម្មភាពរបស់វាដដែលៗ ដោយគោរពតាម៖ថេរពិសេសណាមួយនៅពេលដែល Control Variable នៅតែពិត ។

៤.១ .While Loop Statement

ជាដំបូង While Loop បានត្រួតពិនិត្យមើល Expression បើសិនវាពិត នោះ Statement របស់ While Loop ត្រូវបានអនុវត្តន៍ (Execute) រួចវាក៏ទៅពិនិត្យមើល Expression ម្តងទៀត បើសិនជាវានៅតែពិត នោះ Statement ឬ Block Statement របស់វានៅតែ Execute ដដែល រហូតដល់ Expression មិនពិត ។

Syntax of while loop

```
while (test expression)
{
    statements to be executed.
}
```



ឧទាហរណ៍ ១

Example of while loop

Write a C program to find the factorial of a number, where the number is entered by user. (Hints: factorial of $n = 1*2*3*...*n$)

```
1.
/*C program to demonstrate the working of while loop*/
#include <stdio.h>
int main(){
    int number,factorial;
    printf("Enter a number.\n");
    scanf("%d",&number);
    factorial=1;
    while (number>0){      /* while loop continues until test condition
number>0 is true */
        factorial=factorial*number;
        --number;
    }
    printf("Factorial=%d",factorial);
    return 0;
}
```

Output

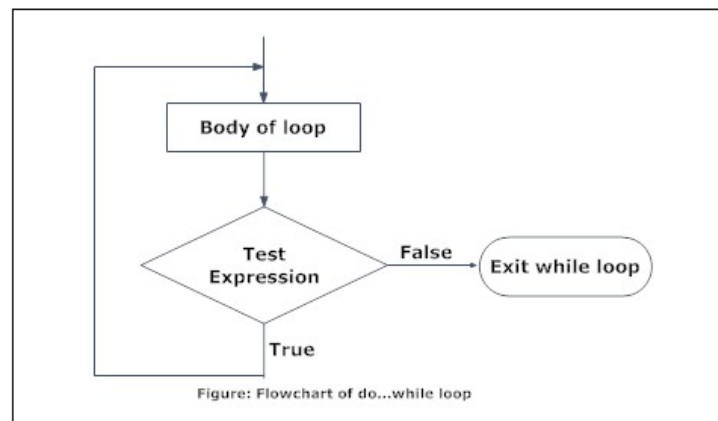
```
Enter a number.
5
Factorial=120
```

៤.២ .Do-While- Loop Statement

ជាដំបូង do While Loop បាន Execute មួយ Statement ដែលនៅមានក្នុង Loop សិនមុននឹងវាត្រួតពិនិត្យមើល Expression ។ ពេលវាត្រួតពិនិត្យឃើញថា Expression នៅតែពិត នោះ Statement ត្រូវបាន Execute ម្តងទៀតជាបន្តបន្ទាប់ រហូតដល់វាពិនិត្យឃើញថា Expression លែងពិត ។

Syntax of do...while loops

```
do {
    some code/s;
}
while (test expression);
```



ឧទាហរណ៍ ១ ៖

Example of do...while loop

Write a C program to add all the numbers entered by a user until user enters 0.

```
1.
/*C program to demonstrate the working of do...while statement*/
#include <stdio.h>
int main(){
    int sum=0,num;
    do          /* Codes inside the body of do...while loops are at
least executed once. */
    {
        printf("Enter a number\n");
        scanf("%d",&num);
        sum+=num;
    }
    while(num!=0);
    printf("sum=%d",sum);
    return 0;
}
16.
```

Output

```
Enter a number
3
Enter a number
-2
Enter a number
0
sum=1
```

ឧទាហរណ៍ ២ ៖

```
#include <stdio.h>

main()
{
    int i = 10;

    do{
        printf("Hello %d\n", i );
        i = i -1;
    }while ( i > 0 );
}
```

This will produce following output:

```
Hello 10
Hello 9
Hello 8
Hello 7
Hello 6
Hello 5
Hello 4
Hello 3
Hello 2
Hello 1
```

ឧទាហរណ៍ ៣ ៖

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

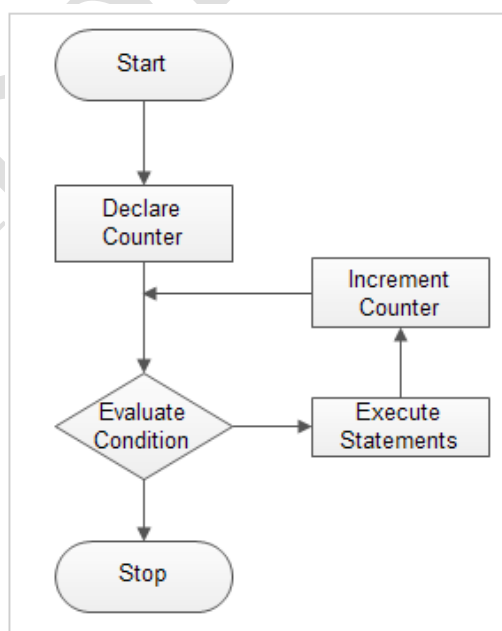
1
2
3
4
5

៤.៣ .For Loop Statement

ទំរង់ For loop មានបីកន្សោម ១ កន្សោមទី មួយត្រូវបានប្រើសំរាប់កំណត់តម្លៃដំបូង កន្សោមទី ២ ប្រើសំរាប់ ផ្ទៀងផ្ទាត់ថា តើ Loop នឹងត្រូវបន្តម្តងទៀត ឬ ទេ ហើយកន្សោមទី ៣ សំរាប់ប្តូរតម្លៃដើម្បីប្រតិបត្តិបន្ត ។

ទំរង់របស់ For Loop ៖

```
for(initial expression; test expression; update expression)
{
    code/s to be executed;
}
```



ឧទាហរណ៍ ១ ៖

```
#include <stdio.h>

void main()
{
    int i = 0, j = 8;
    printf("Times 8 Table\n");
    for(i = 0; i <= 12; i = i + 1)
    {
        printf("%d x %d = %d\n", i, j, j*i);
    }
    printf("\n");
}
```

លទ្ធផល ៖

```
Times 8 Table
0 x 8 = 0
1 x 8 = 8
2 x 8 = 16
3 x 8 = 24
4 x 8 = 32
5 x 8 = 40
6 x 8 = 48
7 x 8 = 56
8 x 8 = 64
9 x 8 = 72
10 x 8 = 80
11 x 8 = 88
12 x 8 = 96
```

ឧទាហរណ៍ ២ ៖

```
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 int main()
11 {
12     int i, k, levels, space;
13     printf("Enter the number of levels in pyramid:");
14     scanf("%d", &levels);
15
16     space = levels;
17
18     for ( i = 1 ; i <= levels ; i++ )
19     {
20         for ( k = 1 ; k < space ; k++ )
21             printf(" ");
22         space--;
23
24         for ( k = 1 ; k <= 2*i - 1 ; k++ )
25             printf("*");
26         printf("\n");
27     }
28     return 0;
29 }
```

```
1  Enter the number of levels in pyramid:10
2
3      *
4     ***
5    *****
6   *********
7  ***********
8  *************
9  *************
10  *************
11  *************
```



ឧទាហរណ៍ ៣ ៖

```

1  #include <stdio.h>
2
3  typedef unsigned long long ULL;
4
5  ULL factorial(const int);
6  void pascal_triangle(const int);
7
8  int main()
9  {
10     int rows;
11
12     printf("--- C Pascal's Triangle Demo ---\n");
13     printf("Enter the number of rows for the Pascal's triangle:");
14     scanf("%d",&rows);
15
16     pascal_triangle(rows);
17
18     return 0;
19 }
20 /*
21  calculate factorial of n
22 */
23 ULL factorial(const int n)
24 {
25     int i;
26     ULL f = 1;
27     for(i = 1 ; i <= n ; i++ )
28         f = f * i;
29     return f;
30 }
31
32 /*
33  displays Pascal's triangle based on
34  the number of rows (rows)
35 */
36 void pascal_triangle(const int rows)
37 {
38     int i, j;
39     ULL k;
40     for ( i = 0 ; i < rows ; i++ )
41     {
42         for(j = 0 ; j <= ( rows - i - 2 ) ; j++ )
43             printf(" ");
44
45         for(j = 0 ; j <= i ; j++ )
46             {

```

លទ្ធផល ៖

```

1  --- C Pascal's Triangle Demo ---
2  Enter the number of rows for the Pascal's triangle:8
3
4      1
5     1 1
6    1 2 1
7   1 3 3 1
8  1 4 6 4 1
9  1 5 10 10 5 1
10 1 6 15 20 15 6 1
11 1 7 21 35 35 21 7 1

```



ឧទាហរណ៍ ៤ ៖ យកតម្លៃខ្លួនឯង គុណ ខ្លួនឯង

```
/* Print out the squares of the first 10 integers */
#include <stdio.h>

void main()
{
    int i;
    for(i = 0; i <= 10; i++)
        printf("%d ", i*i);
    printf("\n");
}
```

```
0 1 4 9 16 25 36 49 64 81 100
Press any key to continue . . .
```

ឧទាហរណ៍ ៥ ៖ កើនតម្លៃតាមការបញ្ចូល

```
#include <stdio.h>

void main()
{
    int i, start = 0, step_by = 0, stop = 0;
    printf_s("Enter three integers: ");
    scanf_s("%d %d %d", &start, &step_by, &stop);
    for(i = start; i <= stop; i = i + step_by)
        printf("%d ", i);
    printf("\n");
}
```

```
Enter three integers: 10 2 20
10 12 14 16 18 20
Press any key to continue . . .
```

៤.៤ .Goto Statement

ប្រើសំរាប់ចាកចេញ បញ្ចូល បង្វែរត្រួតពិនិត្យ ទៅលើ Statement ណាមួយយ៉ាងឆាប់រហ័ស ។

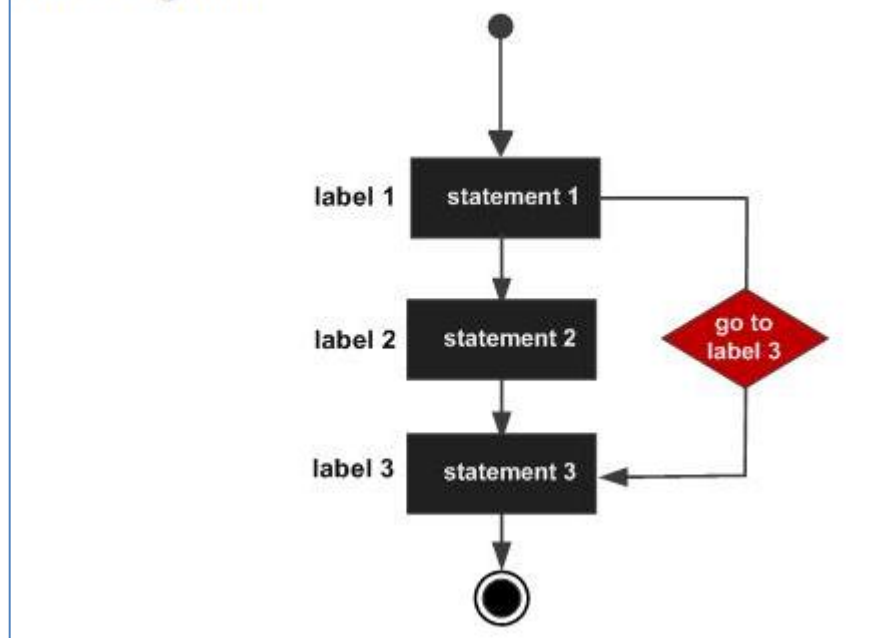
ទំរង់ទូទៅនៃ Goto label

identifier គឺជាឈ្មោះសំគាល់ទីតាំងសំរាប់ Goto statement ទៅ Execute នូវ Statement ដែលនៅបន្ទាប់នឹងវា ។
សូមពិនិត្យមើលខាង

Syntax of goto statement

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

Flow Diagram:



Example:

```

#include <stdio.h>

int main ()
{
    /* local variable definition */
    int a = 10;

    /* do loop execution */
    LOOP:do
    {
        if( a == 15)
        {
            /* skip the iteration */
            a = a + 1;
            goto LOOP;
        }
        printf("value of a: %d\n", a);
        a++;
    }while( a < 20 );

    return 0;
}

```

```

value of a: 10
value of a: 11
value of a: 12
value of a: 13
value of a: 14
value of a: 16
value of a: 17
value of a: 18
value of a: 19

```

ចប់បេឡេនទី ៤

មេរៀនទី ៥ ៖ Function

៥.១ .សេចក្តីផ្តើម

ដើម្បីដោះស្រាយបញ្ហាធំ ហើយស្មុគស្មាញ យើងត្រូវបំបែកបញ្ហាអោយទៅជាបញ្ហាតូចៗជាងមុនសិន ទើបយើងអាចដោះស្រាយបញ្ហាធំៗនោះបាន ។ ក្នុងការសរសេរកម្មវិធី គឺយើងមិនអាចសរសេរដោយ ពុំបាន បានបំបែកកម្មវិធីនោះ ទៅជាកម្មវិធីតូចៗទេ ព្រោះការបំបែកកម្មវិធីនេះ នាំអោយមានការលំបាក ការកែរនៅពេល កម្មវិធីមានភាព Error និងពិបាកក្នុងការសរសេរ ។ Function មានដូចជា ៖

- sub prigram
- វាមានលក្ខណៈងាយស្រួលក្នុងការគ្រប់គ្រង
- ងាយស្រួលកែនៅពេលមានភាព Error កើតឡើង
- កាត់បន្ថយទំហំ Memory
- ធ្វើអោយកម្មវិធីដំណើរការមានប្រសិទ្ធភាព

៥.២ .លក្ខណៈទូទៅ របស់ Sub Program or Function

នៅក្នុងភាសា C ឬ C++ Function ត្រូវបានចែកចេញជា ពីរ គឺ ៖

- Standard Library Function
- Function បង្កើតឡើងដោយ Programmer

1... Standard Library Function

ជាប្រភេទអនុគមន៍ដែលគេសរសេររួចជាស្រេចនិង បញ្ចូលទៅក្នុង Computer នៃកម្មវិធី ។ ការហៅ Function ទាំងនេះមកប្រើ គឺទៅតាមរយៈ Header file ។ Standard Library Function គេមិនអាចកែប្រែបានទេ ។ ដែលយើងសិក្សាដូចជា ៖ Printf , scanf , goto;

Function មានខ្លះដូចខាងក្រោម ៖

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

ឧទាហរណ៍ ១ ៖

```
/* Program of calculates Addition's two constant */
#include<stdio.h>
main() {
    void sum(void);
    char c;
    c = getchar( );
    while ( c != EOF ) {
        sum( );
        c = getchar( );
    }
    return ( 0 );
}

void sum( void ) {
    int x, y;
    clrscr( );
    printf( "Input x, y : " );
    scanf( "%d %d", &x, &y );
    printf( "sum = %d", x + y );
}
```



៥.៣ .Function Declaration

គឺជាការប្រកាស Function មុនពេលយើងហៅ Function នោះមកប្រើ ។ គេច្រើនប្រកាសវានៅលើ ឬ នៅក្នុង Main() Function ។ ការប្រកាសនេះដើម្បីប្រាប់ Compiler ថា Code របស់ Function Declaration មាននៅខាងក្រោម ។ Function Declaration ត្រូវបានបញ្ចប់ដោយប្រើ Semicolon (;)

ឧទាហរណ៍យើងសរសេរកម្មវិធីគណនាផលបូកមួយដោយប្រើប្រាស់ Function ដូចខាងក្រោម ៖

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

/* Function Declaration */
int sum ( int, int );

main( )
{
    int x, y, s;
    scanf ( "%d %d", &x, &y );
    s = sum ( x, y );           // x and y called Argument
    printf ( "sum = %d", s );
    return;
}

/* Function Definition */

int sum ( int a, int b )       // a and b called Parameter
{
    return a + b;
}
```

ចំណាំ ៖ Parameter និង Argument មិនចាំបាច់មានឈ្មោះដូចគ្នាទេ ។

៥.៤ .អនុគមន៍ Call អនុគមន៍

បើសិនជាអនុគមន៍ Call អនុគមន៍ នោះគេមានមធ្យោបាយពីរក្នុងការប្រកាសអនុគមន៍ ៖

- ១ ការប្រកាសអនុគមន៍នៅក្នុង Main(): ការប្រកាសបែបនេះគឺគេហៅវាថា Local ប៉ុណ្ណោះ មានន័យថាយើងអាចប្រើ Function នោះបានតែនៅក្នុង Main() មិនអាចប្រើនៅកន្លែងណាបានទេ ។
- ២ ការប្រកាសអនុគមន៍នៅក្នុង Main(): ការប្រកាសបែបនេះគឺគេហៅវាថា External Function គឺអោយ Function នោះអាចប្រើនៅគ្រប់ទីកន្លែងផ្សេងទៀតនៅក្នុងកម្មវិធីទាំងមូល ។

៥.៥ .Return Statement

ជាធម្មតា C program គឺតែងតែមានការប្រើនូវ Return Statement ព្រោះ C program ចាំបាច់ត្រូវតែមាន Function មួយជានិច្ចដូចជា ៖ main() Function ជាដើម ។ return statement ត្រូវបានប្រើសំរាប់បញ្ចប់ការ Execute នៅក្នុង Function ទាំងអស់ ។ ប៉ុន្តែពេលដែល Function ពុំមានតម្លៃត្រូវ Return ទេ ។



៥.៦ .សំណត់

សំណត់

សូមចូលសញ្ញា ☒ តើអ្នកស្គាល់ច្បាស់ពីច្រើនត្រូវ តែបើមិនត្រូវសូមចូលសញ្ញាខ្សែ(ឧ) នូវច្បាប់ត្រូវនេះ

- 1- ☐ C Program ចាំបាច់ត្រូវតែមាន Function មួយជាតិច ។
- 2- ☐ បើសិនជា C Program មានពីរ Function : main និង mystery ។ mystery function ចាំបាច់ត្រូវតែប្រើនៅក្នុង main Function ។
- 3- ☐ ការហៅប្រើ Function ដាច់ខាតត្រូវតែមាន Arguments ។
- 4- ☐ Argument និង Parameter ដែលកំពុងទាក់ទងគ្នា Parameter ត្រូវតែមានឈ្មោះដូច Argument ។
- 5- ☐ រាល់ Function ទាំងអស់តែងតែ Return Value ពេលប្រើប្រាស់វា ។
- 6- ☐ Parameter ត្រូវបានប្រកាសនៅក្នុង Function's Header ។
- 7- ☐ Function មួយ ត្រូវតែមាន return Statement លើសពីមួយ ។
- 8- ☐ Function មួយ អាច return តម្លៃលើសពីមួយតម្លៃ នៅក្នុងពេលតែមួយ ។
- 9- ☐ return Statement តែងតែមានប្រើប្រាស់នៅក្នុង Function មួយ ។
- 10- ☐ វាជាការត្រឹមត្រូវ ដែលយើងសរសេរ Syntax បែបនេះ : return;
- 11- ☐ នៅក្នុង C អនុញ្ញាតឱ្យអ្នកប្រើប្រាស់ផ្សេងគ្នាមានឈ្មោះដូចគ្នា ។
- 12- ☐ យើងអាចហៅ Function មួយមកប្រើជាមួយ call Statement ដូចខាងក្រោមនេះ :
- 13- ☐ យើងមាន code ដូចខាងក្រោម :

```
z = 111;
fun( ++z, ++z, ++z);
```

fun មាន Argument បី ដែលនឹងគណនាបានតម្លៃ 112, 113, និង 114 ជាលំដាប់ ។
- 14- ☐ នៅពេលដែល Function ត្រូវបានហៅមកប្រើ គ្រប់ Argument ទាំងអស់របស់វាត្រូវបានគណនាមុននឹងបញ្ជូនទៅអោយ Function's Parameter ។

ចប់មេរៀនទី ៥

មេរៀនទី ៦ ៖ Array

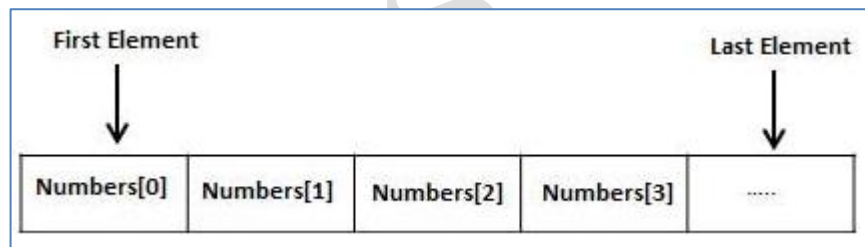
៦.១ និយមន័យ

Array គឺជាសំនុំនៃ Variable ដែលមាន Data type ដូចគ្នាដែលមានឈ្មោះតែមួយ។ ធាតុនីមួយៗរបស់ Array គឺ Variable 1 បើមានការកែប្រែទៅលើធាតុនោះ ធាតុដទៃគ្មានការប្រែប្រួលឡើយ ។

- គេសំគាល់ធាតុរបស់ Array ដោយ Index ឬក៏ Subscripts ដែលមានធាតុទី ១ មាន Index=0 រៀងគ្នារហូតដល់ទី n ដែលមាន Index = n-1
- ទំរង់របស់ Array ត្រូវតែជាចំនួនគត់ ហើយធំជាងសូន្យ

Data-Type Array_Name[Length];

- Data type គឺជាប្រភេទទិន្នន័យ
- Array_Name គឺជាឈ្មោះរបស់ Array
- Length គឺជាប្រវែង ឬ ទំហំរបស់ Array



Ex 9 .

```
#include <stdio.h>

int main ()
{
    int n[ 10 ]; /* n is an array of 10 integers */
    int i,j;

    /* initialize elements of array n to 0 */
    for ( i = 0; i < 10; i++ )
    {
        n[ i ] = i + 100; /* set element at location i to i + 100 */
    }

    /* output each array element's value */
    for ( j = 0; j < 10; j++ )
    {
        printf("Element[%d] = %d\n", j, n[j] );
    }

    return 0;
}
```



លទ្ធផល

```

Element[0] = 100
Element[1] = 101
Element[2] = 102
Element[3] = 103
Element[4] = 104
Element[5] = 105
Element[6] = 106
Element[7] = 107
Element[8] = 108
Element[9] = 109

```

Ex ២ .

```

#include<stdio.h>

int main()
{
    // Declaring/Initializing three characters pointers
    char *ptr1 = "Himanshu";
    char *ptr2 = "Arora";
    char *ptr3 = "TheGeekStuff";

    //Declaring an array of 3 char pointers
    char* arr[3];

    // Initializing the array with values
    arr[0] = ptr1;
    arr[1] = ptr2;
    arr[2] = ptr3;

    //Printing the values stored in array
    printf("\n [%s]\n", arr[0]);
    printf("\n [%s]\n", arr[1]);
    printf("\n [%s]\n", arr[2]);

    return 0;
}

```

លទ្ធផល

```

$ ./charptrarr

[Himanshu]

[Arora]

[TheGeekStuff]

```

៦.២ Two Dimensional Array

កម្មវិធី C បានផ្តល់អោយមានការប្រើប្រាស់នូវ Array ពីរវិមាឌ (Two dimension arrays) ប៉ុន្តែវាត្រូវបានគេប្រើប្រាស់តិចជាង Array មួយវិមាឌ ។

Syntax:

Data-Type Array_Name[Length 1] [Length 2];

ឧទាហរណ៍ ៖ យើងប្រកាស Array ពីរវិមាឌដូចខាងក្រោម ៖

```
float dr [4] [5];
```

បញ្ជាក់ថាមាន ២៤ Variable ជាប្រភេទ float ដូចខាងក្រោមបង្ហាញក្នុងរូបខាងក្រោម ៖

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

៦.៣ Array and Function

ឧបមាថាយើងមាន Array ធាតុមួយជា Int ដែលមាន 10 ធាតុ(Element) គឺ Int(10)។ យើងត្រូវប្រើប្រាស់បង្ហាញធាតុ ឬ វាយបញ្ចូលធាតុរបស់ Array ដោយប្រើប្រាស់អនុគមន៍ ។ តើយើងត្រូវរៀបចំអនុគមន៍ដោយរបៀបណា ? តើអនុគមន៍នោះមាន Parameter ដែរឬទេ ? ហើយ Parameter នោះជា Array ឬតម្លៃអថេរផ្សេងទៀតឬយ៉ាងណា ?

គឺយើងអាចធ្វើការប្រកាស Function ដែល Argument របស់វាជា Array មានធាតុ ១០ ដូចខាងក្រោម ៖

1. void Array_Function (int b[10]);
2. void Array_Function (int b [], int n)

ពេលយើងហៅ (Call) Function មកប្រើត្រូវ សរសេរដូចខាងក្រោម

1. Array_Function(b)
2. Array_Function(b,n)

ឧទាហរណ៍ ៣ :

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

/* Declaration Function has Argument Array */
void funct( int a[ 10 ] );
/* or write: void funct( int a[ ] ); */
/* or write: void funct( int [ ] ); */

main( ) {
    int value[ 10 ] = { -30, 72, 100, 10, 5, 21, 22, 100, 17, 35 };
    clrscr( );
    funct( value );      /* Call Function Array */
    printf("In Press <q> to Exit ! ");
    while( getch( ) != 'q' );
    return 0;
}

void funct( int ar[ 10 ] ) {
    int i;
    for( i=0; i<10; i++ )
        printf( "Array [%d]=%3d \n", i, ar[ i ] );
    putchar( '\n' );
}
```



៦.៤ Array String

នៅក្នុង C គ្មានប្រភេទទិន្នន័យជា String ដូចនៅក្នុងភាសាទីដទៃទេ ។ ដូច្នេះភាសា C String ត្រូវបានជំនួសមកវិញនូវ Array នៃ Character ។ គេអាចនិយាយបានថា String គឺជាសេរីនៃតួអក្សរតៗគ្នាដែលប្រើប្រាស់ Memory Location ឬគេនិយាយបានថា String គឺជា Array នៃ Array ដែលមាន Dimension កំណត់នូវទីតាំង ឬ Location របស់ Memory ។ ជាទូទៅគ្រប់ String ទាំងអស់ត្រូវបានបញ្ចប់ដោយ Null ('\n') Charater ។

ឧទាហរណ៍ Char str[10]="Computer"

មានន័យថា ៖

str[0] = 'C';	str[5] = 't';
str[1] = 'o';	str[6] = 'e';
str[2] = 'm';	str[7] = 'r';
str[3] = 'p';	str[8] = '\0';
str[4] = 'u';	str[9] = '\0';

Memory:

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
'C'	'o'	'm'	'p'	'u'	't'	'e'	'r'	'\0'	'\0'

៦.៥ លំហាត់

លំហាត់

1. តើ Array ខាងក្រោមមានអថេរប៉ុន្មាន?
`int trucks_in_stock[25];`
 2. បើសិនជាយើងផ្លាស់ Array ខាងលើពី ២៥ ទៅដាក់ ៥៥ វិញ តើវាមានអថេរប៉ុន្មាន?
 3. តើរាល់អថេររបស់ Variables ខាងលើ ជាប្រភេទអ្វី?
 4. ហេតុអ្វីបានជាអ្នកគិតថា អ្នកបង្កើតភាសា C ជ្រើសរើសយកសញ្ញា Brackets (" [" និង "] ") ហើយមិនប្រើសញ្ញាវង់ក្រចក " (" និង ") " ជាមួយ Array ?
- យើងបានប្រកាស Array មួយដូចខាងក្រោម ៖
- ```
char letters[26];
```
5. (T/F) Array letters មាន ២៦ ធាតុ។
  6. តើ letters[ 1 ] ត្រូវគ្នានឹងធាតុរបស់ Array ខាងលើណាមួយ?
  7. តើ letters[ 26 ] មាននៅក្នុងធាតុរបស់ Array ខាងលើដែរឬទេ?
  8. ឧបមាថាយើងបានដាក់អក្សរ A ក្នុង letters[0] បន្ទាប់មក B ក្នុង letters[1] និងជាបន្តបន្ទាប់រហូតដល់ Z ក្នុង letters[25] ។ តើយើងមានលទ្ធផលបែបណាចំពោះ code ខាងក្រោមនេះ៖  

```
temp = letters[25];
letters[25] = letters[0];
letters[0] = temp;
```
  9. តើ Array letters មានប៉ុន្មានធាតុ?
  10. តើ Index ត្រឹមត្រូវរបស់ Array letters ប៉ុន្មាន?

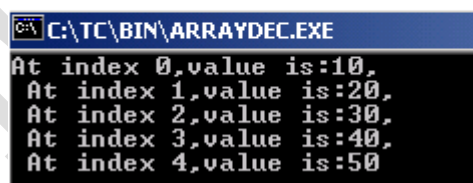


ចូលសាកល្បងធ្វើតេស្ត ថាតើវាចេញលទ្ធផលយ៉ាងម៉េចដែរ

```
#include <stdio.h>
#include <conio.h>
int arr[5];
void main() {
 arr[0]=10;
 arr[1]=20;
 arr[2]=30;
 arr[3]=40;
 arr[4]=50;
 clrscr();
 printf(
 "At index 0,value is:%d,"
 "\n At index 1,value is:%d,"
 "\n At index 2,value is:%d,"
 "\n At index 3,value is:%d,"
 "\n At index 4,value is:%d\n",
 arr[0], arr[1], arr[2], arr[3], arr[4]);
 getch();
}
```

Output will be displayed as:

ARRAYDEC.EXE



```
C:\TC\BIN\ARRAYDEC.EXE
At index 0,value is:10.
At index 1,value is:20.
At index 2,value is:30.
At index 3,value is:40.
At index 4,value is:50
```

ចប់មេរៀនទី ៦



## ❀ មេរៀនទី ៧ ៖ Pointer ❀

### ៧.១ និយមន័យ

Pointer គឺជាប្រភេទទិន្នន័យមួយ ដែលប្រាស្រ័យទៅលើប្រភេទទិន្នន័យដើម ដែលមានតម្លៃជា Address ចំពោះអញ្ញាតដែលយើងប្រើធ្វើការប្រកាសជា Pointer វាអាចផ្ទុកបានតម្លៃជា Address នៃអញ្ញាតដទៃទៀតដែលវាបានចង្អុលទៅកាន់ ហើយអញ្ញាតទាំងពីរត្រូវមានប្រភេទទិន្នន័យដូចគ្នា ។

### ៧.២ របៀបប្រកាស Pointer

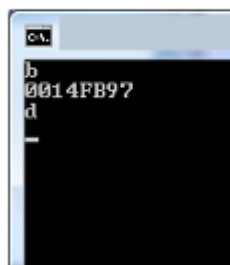
ចំពោះការប្រកាស Pointer មានលក្ខណៈដូចអញ្ញាតធម្មតា តែមានសញ្ញា ( \* ) នៅខាងមុខជានិច្ច ។

```
<variable_type> *<name>;
```

ឧទ.

```
#include <stdio.h>
int main()
{
 char a='b';
 char *ptr;
 printf("%cn",a);
 ptr=&a;
 printf("%pn",ptr);
 *ptr='d';
 printf("%cn",a);
 return 0;
}
```

Output



## ៧.២ Pointer to Pointer

យើងអាចប្រើ Pointer ដើម្បីចង្អុលទៅកាន់អញ្ញាដទៃ ហើយក៏អាចចង្អុលទៅកាន់អញ្ញាជា Pointer មួយផ្សេងបានដែរ ប៉ុន្តែ Pointer ទាំងនោះត្រូវមានកំរិតផ្សេងពីគ្នា ។

ឧទាហរណ៍ ៖

```
#include<stdio.h>
#include<conio.h>
void main()
{
 clrscr();
 int x = 10;
 int *a;
 int **b;
 int ***c;
 a = &x;
 b = &a;
 c = &b;
 printf("value of x = %d\n",x);
 printf("value of x by using a = %d\n",*a);
 printf("value of x by using b = %d\n",**b);
 printf("value of x by using c = %d\n",***c);
 getch();
}
```

### Output:

```
value of x = 10
value of x by using a = 10
value of x by using b = 10
value of x by using c = 10
```

## ៧.២ Pointer to Function

ចំពោះ Pointer យើងអាចប្រើជា Address នៃអនុគមន៍ផងដែរ ដោយវាចាប់យក Address របស់អនុគមន៍ណាមួយសំរាប់ប្រើប្រាស់ ក្នុងអនុគមន៍

ឧទាហរណ៍ ១ ៖

### Output:

```
Before calling function.
a = 10
b = 20
After calling function.
a = 10
b = 70
```

```
#include<stdio.h>
#include<conio.h>
void change(int,int*);
void main()
{
 clrscr();
 int a = 10 , b = 20;
 printf("Before calling function.\n");
 printf("a = %d \n",a);
 printf("b = %d \n",b);
 change(a,&b);
 printf("After calling function.\n");
 printf("a = %d \n",a);
 printf("b = %d \n",b);
 getch();
}
void change(int x,int *y)
{
 x = x * 5;
 *y = *y + x;
}
```

ឧទាហរណ៍ ២ ៖

```
#include<stdio.h>
#include<conio.h>
int (*psum)(int,int);
int sum(int,int);
void main()
{
 clrscr();
 int a = 40 , b = 20 ;
 psum=∑
 printf("Result = %d", (*psum) (a,b));
 getch();
}

int sum(int m , int n)
{
 int s;
 s = m + n ;
 return (s);
}
```

Output:

Result = 60

❖ចប់មេរៀនទី ៧❖



## មេរៀនទី ៨ ៖ File

### ៨.១ និយមន័យ

គឺជាឯកសារ ឬ បណ្តុំទិន្នន័យទាំងឡាយណាដែលអាចរក្សាទុកនៅក្នុង ឧបករណ៍ផ្ទុកព័ត៌មាន ដូចជា Hard Disk , Flash ជាដើម ។ នៅក្នុងភាសា C Program File គឺជា Byte ដែលរៀបគ្នាជាបន្តបន្ទាប់គ្នាជាស្វ័ត ប្រើដើម្បីផ្ទុកទិន្នន័យតាមពីទំរង់ គឺ Text File និង Binary File ។

### ៨.២ របៀបប្រកាស File

ដើម្បីប្រើប្រាស់ទិន្នន័យនៅក្នុង file ត្រូវមាន file pointer មួយសំរាប់ចង្អុលទៅកាន់ទីតាំងណាមួយដែលយើងត្រូវការដើម្បី read / write ទិន្នន័យ ។

Syntax: **FILE \* identifier ;**

### ៨.៣ របៀបប្រកាស Text File

គឺជា File ដែលផ្ទុកទិន្នន័យក្រោមទំរង់ Graphic Character មានន័យថាគ្រប់ធាតុរបស់ File ទាំងអស់ត្រូវបានបំប្លែងទៅជាតួអក្សរដែលអាចមើលឃើញ ។ ការបញ្ចូលទិន្នន័យ ឬ ទាញទិន្នន័យនៃ File យើងអាចប្រកាស File Pointer សំរាប់ចង្អុលទៅកាន់ធាតុរបស់ File មួយដែលមានឈ្មោះជាក់លាក់ ដែលត្រូវបើ ឬ បង្កើតឡើងដោយអនុគមន៍ fopen() ។

រូបមន្ត

```
fp=fopen("FileName","Model")
```

-FileName : គឺជាឈ្មោះរបស់ File ដែលមាន FileName និង Execute ដែលយើងត្រូវប្រើប្រាស់

-Model : គឺជា String សំរាប់កំណត់ពីគោលបំណងនៃការបើដែលមានលក្ខណៈដូចខាងក្រោមនេះ

| Mod    | អត្ថន័យ                                                         |
|--------|-----------------------------------------------------------------|
| r/rt   | បើ Text File សំរាប់អានទិន្នន័យ , បើគ្មាន File វានឹង error       |
| w/wt   | បើ Text File សំរាប់ផ្ទុកទិន្នន័យ , បើគ្មាន File វានឹងបង្កើតថ្មី |
| a/at   | បើ Text File សំរាប់បន្ថែមទិន្នន័យ,បើគ្មាន Fileវានឹងបង្កើតថ្មី   |
| r+/r+t | បើ Text File សំរាប់បញ្ចូលទិន្នន័យ ឬ អានទិន្នន័យ                 |
| w+/w+t |                                                                 |
| a+/a+t |                                                                 |

### ៨.៣.១ បើក File សំរាប់ផ្ទុក

អនុគមន៍ខាងក្រោមនេះត្រូវបានប្រើសំរាប់ផ្ទុកទិន្នន័យទៅក្នុង file មួយ ។ អនុគមន៍ putc(), fputc(); ត្រូវបានគេប្រើសំរាប់បញ្ចូលទិន្នន័យមួយតួអក្សរទៅក្នុង File តាមរយៈ File Pointer ។

```
#include<stdio.h>
#include<conio.h>
void main()
{
 clrscr();
 char ch;
 FILE *fp;
 fp=fopen("Test1.txt","w");
 if(fp==NULL)
 printf("Opening file was Error");
 else
 { printf("Input character store in file.\n");
 printf("Press <Esc to stop>\n");
 do
 {
 ch=getch();
 if(ch!=27)
 {
 putchar(ch);
 fputc(ch,fp);
 }
 }
 while(ch!=27);
 }
 fclose(fp);
}
```

### ៨.៣.២ បើក File សំរាប់អាន

អនុគមន៍ខាងក្រោមនេះត្រូវបានប្រើសំរាប់អានទិន្នន័យទៅក្នុង file មួយ ។ អនុគមន៍ `getc()`, `fgetc()`;  
ត្រូវបានគេប្រើសំរាប់ទាញទិន្នន័យមួយតួអក្សរទៅក្នុង File តាមរយៈ File Pointer ។

```
#include<stdio.h>
#include<conio.h>
void main()
{ clrscr();
char ch;
FILE *fp;
fp=fopen("Test1.txt","r");
if(fp==NULL)
printf("Opening file was Error.");
else
{ printf("Read data from File\n");
ch=getc(fp);
while(ch!=EOF)
{ putchar(ch);
ch=getc(fp);
}
}
fclose(fp);
getch();
}
```



### ៨.៣.៣ បើក Text File សំរាប់បន្ថែមទិន្នន័យ

អនុគមន៍ខាងក្រោមនេះត្រូវបានប្រើសំរាប់បន្ថែមទិន្នន័យទៅក្នុង file មួយ ។ អនុគមន៍ putc(), fputc(); ត្រូវបានគេប្រើសំរាប់បន្ថែមទិន្នន័យមួយតួអក្សរទៅក្នុង File តាមរយៈ File Pointer ។

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

void main()
{ clrscr();
 char ch;
 FILE *fp;
 fp=fopen("Test1.txt","a");
 if(fp==NULL)
 printf("Opening file was Error");
 else
 { printf("Input character to append in
 file.\n");
 printf("Press <Esc to stop>\n");
 do
 { ch=getch();
 if(ch!=27)
 { putchar(ch);
 fputc(ch,fp);
 }
 }while(ch!=27);
 }
 fclose(fp);
}
```



## ៨.៤ Binary File

Binary File ដែលផ្ទុកទិន្នន័យក្រោមទម្រង់ Internal Format មានន័យថាគ្រប់ធាតុរបស់ File ទាំងអស់ត្រូវបានរក្សាទុកទិន្នន័យជា int , long , double , float ជាដើម ។ រីឯការបញ្ចូលទិន្នន័យ ឬ ទាញទិន្នន័យនៃ File មានលក្ខណៈ ដូច Text File ដែរ ប៉ុន្តែអាស្រ័យដោយ mode របស់វា ។

Ex:   File \*fp;  
       fp = fopen("FileName","Mode");

| Mode | Meaning                                                             |
|------|---------------------------------------------------------------------|
| rb   | បើក Binary File សំរាប់អានទិន្នន័យ, បើគ្មាន File វានឹងផ្តល់ការ Error |
| wb   | បើក Binary File សំរាប់ផ្ទុកទិន្នន័យ, បើគ្មាន File វានឹងបង្កើតថ្មី   |
| ab   | បើក Binary File សំរាប់បន្ថែមទិន្នន័យ, បើគ្មាន File វានឹងបង្កើតថ្មី  |
| r+b  | បើក Binary File សំរាប់បញ្ចូល រឺ អានទិន្នន័យ                         |
| w+b  |                                                                     |
| a+b  |                                                                     |

## ចប់មេរៀនទី ៨



Kompongsom4u



Komongsom4u



Kompongsom4u



Kompongsom4u



Kompongsom4u



Kompongsom4u

