

មេរៀនទី១

ការណែនាំពីមូលដ្ឋានគ្រឹះ

របស់រចនាសម្ព័ន្ធទិន្នន័យ និងវិធីសាស្ត្រ

មេរៀននេះណែនាំឱ្យស្វែងយល់អំពីបញ្ហារចនាសម្ព័ន្ធទិន្នន័យ និង វិធីសាស្ត្រ (Data Structure and Algorithms) និងអំពីគ្នាទៅវិញទៅមកនៃទំនាក់ទំនងរបស់វានៅក្នុងកម្មវិធី និងការស្វែងយល់អំពីភាសា (Structure language) ដែលបានយកមកប្រើប្រាស់សម្រាប់អនុវត្តន៍លើរចនាសម្ព័ន្ធ (Structure) របស់វិធីសាស្ត្រ (Algorithms) ។

រាល់ចំណោទបញ្ហា ដែលត្រូវដោះស្រាយជាមូលដ្ឋាននៃភាសាម៉ាស៊ីន ចាំបាច់ត្រូវស្វែងយល់អំពី រចនាសម្ព័ន្ធទិន្នន័យ និងវិធីសាស្ត្រ (Data structure and Algorithms) ជាមុនសិន ពីព្រោះរចនាសម្ព័ន្ធទិន្នន័យ និងវិធីសាស្ត្រ (Data structure and Algorithms) ជាបញ្ហាមូលដ្ឋាននៃការសរសេរកម្មវិធី (Programming) ។ យើងមានរចនាសម្ព័ន្ធ (Structure) ដែលបានបង្កើតឡើងដោយលោក Nicklaus Wirth បានសរសេរ រចនាសម្ព័ន្ធទិន្នន័យ (Data structure) + វិធីសាស្ត្រ (Algorithms) = កម្មវិធី (Program) មានន័យថា៖

ចំណោទបញ្ហា → រកដំណោះស្រាយ និងរៀបចំរបៀបដោះស្រាយ → សរសេរកម្មវិធី។

កម្មវិធី (Program) គឺជាសំនុំមួយសម្រាប់ផ្ទុកនូវរាល់គំនិតយោបល់របស់អ្នកសរសេរកម្មវិធីទៅតាមប្រព័ន្ធកំណត់ របស់ម៉ាស៊ីន ដែលក្រោយពីការអនុវត្តន៍ទៅតាមដំណើរការ (Process) របស់ម៉ាស៊ីនយើងទទួលបានលទ្ធផលដែលចង់បាន។

រចនាសម្ព័ន្ធទិន្នន័យ (Data Structure) គឺជាការរៀបចំទិន្នន័យ (Data) របស់អ្នកសរសេរកម្មវិធី (Programmer) ទៅតាមកូដ (Code) របស់ម៉ាស៊ីន និងរាល់ទិន្នន័យ (Data) ដែលយើងផ្តល់ឱ្យ រួចទាញយកមកប្រើប្រាស់វិញក្នុងគោលបំណងអ្វីមួយ។

វិធីសាស្ត្រ/អាល់ហ្គោរីត (Algorithms) គឺជាជំហាន នៃការដោះស្រាយបញ្ហាទៅតាមដំណាក់កាលៗ ជាតំនូសតាងលំហូរ (Flowchart) និងម៉ូឌុលទៀតវិធីសាស្ត្រ (Algorithms) គឺជាខ្សែបកាសន៍ (Statement) ដ៏ច្បាស់លាស់ដែលក្រោយពីការកំណត់ជាក់លាក់ យើងទទួលបានលទ្ធផលដែលយើងចង់បាន។ យើងមានរចនាសម្ព័ន្ធ (Structure) ទូទៅសម្រាប់ស្វែងយល់បញ្ហាបន្ថែមទៀតដែលត្រូវបានបង្កើតឡើងដោយលោក Kowalski ដែលជាអ្នកបង្កើតកម្មវិធីតក្កវិទ្យា (Logic) ត្រូវបានសរសេរ៖ វិធីសាស្ត្រ (Algorithms) = តក្កវិទ្យា (Logic) + ត្រួតពិនិត្យ (Control)

តក្កវិទ្យា (Logic) គឺជាចំណេះដឹងមូលដ្ឋានប្រើសម្រាប់កំណត់អត្ថន័យឱ្យ Algorithms ផ្នែក Logic ជាអ្នកដឹង Algorithms ត្រូវធ្វើ។

ត្រួតពិនិត្យ (Control) គឺជាវិធីសាស្ត្រដោះស្រាយលើផ្នែកតក្កវិទ្យា (Logic) វាឱ្យដឹងថាវិធីសាស្ត្រ (Algorithms) ត្រូវធ្វើដូចម្តេច។

សូមពិនិត្យមើលនូវឧទាហរណ៍ដូចខាងក្រោម៖

ចូរលើកឡើងអំពីលក្ខណៈតក្កវិទ្យា និងត្រួតពិនិត្យ (Logic and control) របស់ចំណោទបញ្ជា $n!$ ។

យើងអាចកំណត់ដូចខាងក្រោម៖

តក្កវិទ្យា (Logic) របស់បញ្ជា $n! = 1 \times 2 \times 3 \times \dots \times n$

ត្រួតពិនិត្យ (Control) របស់ $n!$ លើផ្នែកតក្កវិទ្យា (Logic) នេះរួមមាន ៣ របៀបដូចខាងក្រោម៖

របៀបទី១៖ គុណបង្រួមបណ្តាលលេខពី ១ ដល់ n ដាក់ចូលក្នុងអថេរណាមួយ

$f = 1;$

for($i = 1; i \leq n; i++$)

$f = f * i;$

របៀបទី២៖ គុណបង្រួមបណ្តាលលេខពី n ដល់ ១ ដាក់ចូលក្នុងអថេរណាមួយ

$f = n;$

for($i = 1; i \geq 1; i--$)

$f = f * i;$

របៀបទី៣៖ គុណបង្រួមបណ្តាលលេខសេសជាមួយគ្នាដាក់ចូលក្នុងអថេរណាមួយ និងគុណបង្រួមបណ្តាលលេខគូជាមួយគ្នាដាក់ចូលក្នុងអថេរណាមួយទៀត បន្ទាប់មកគុណលទ្ធផលទាំងពីរជាមួយគ្នាដាក់ក្នុងអថេរណាមួយ។

២. បញ្ហាទំនាក់ទំនងមួយចំនួនរបស់រចនាសម្ព័ន្ធទិន្នន័យ

រចនាសម្ព័ន្ធទិន្នន័យ (Data structure) សម្តែងដំណើរការនៅក្នុងមេម៉ូរី (Memory) ឱ្យឈ្មោះថា រចនាសម្ព័ន្ធនៃ (Storage Structure)។

កំណត់រចនាសម្ព័ន្ធទិន្នន័យជាមុន (Predefine Data structure) គឺជារចនាសម្ព័ន្ធ (Structure) ដែលមានស្រាប់ នៅក្នុងភាសាកម្មវិធី ដូចជា រចនាសម្ព័ន្ធអ៊ីរ៉េ (Array Structure)។ ការជ្រើសរើសរបៀបកំណត់រចនាសម្ព័ន្ធទិន្នន័យជាមុន (Predefine Data structure) សមស្របសម្រាប់ដំណោះស្រាយបញ្ហា នាំឱ្យមានការដំណើរការវិធីសាស្ត្រ (Algorithms) មានលឿនលឿន និងផ្តល់លទ្ធផលត្រឹមត្រូវ។

រាល់ចំណោទបញ្ជា ដែលមានទ្រង់ទ្រាយតូចមានបរិមាណទិន្នន័យ (Data) តិច និងរាល់ការធ្វើប្រមាណវិធីរបស់វាគ្រាន់តែការប្រើប្រមាណវិធី $+$, $-$, $*$, $/$ និងតក្កវិទ្យា (Logic) ត្រូវបានគេឱ្យឈ្មោះថា បញ្ហាបច្ចេកទេស។

រាល់ចំណោទបញ្ជា ដែលមានទ្រង់ទ្រាយធំមានបរិមាណទិន្នន័យ (Data) ច្រើន ហើយរាល់ការធ្វើប្រមាណវិធីរបស់វាមិនគ្រាន់តែប្រើប្រមាណវិធី $+$, $-$, $*$, $/$ និងតក្កវិទ្យា (Logic) ប៉ុណ្ណោះទេ គ្រឹះត្រូវបានអនុវត្តន៍

ជាមួយប្រមាណវិធីមួយចំនួនទៀត ដូចជា បេក្ខីន, បន្ថែម, បន្ថយ, លុប, ស្វែងរក និងតំរៀបជាដើមឱ្យឈ្មោះថា ចំណោទបញ្ហាមិនមែនជាចំនួន (Non Numerical Problem)។

៣. តួនាទីនិងទំនាក់ទំនងគ្នារវាងរចនាសម្ព័ន្ធទិន្នន័យ និងវិធីសាស្ត្រ

នៅពេលដោះស្រាយជាមួយម៉ាស៊ីន (Machine) ជាដំបូង យើងត្រូវគិតដល់រចនាសម្ព័ន្ធទិន្នន័យ និងវិធីសាស្ត្រ (Data structure and Algorithms) របស់បញ្ហានោះ។

វិធីសាស្ត្រ (Algorithms) មានតួនាទីធ្វើការដំណើរការរបស់បញ្ហា រីឯរចនាសម្ព័ន្ធទិន្នន័យ (Data Structure) មានតួនាទីធ្វើដំណើរការជាមួយម៉ាស៊ីន (Machine) ដើម្បីនាំដល់លទ្ធផលដែលចង់បានរបស់ចំណោទបញ្ហា។

ទំនាក់ទំនងគ្នារវាងរចនាសម្ព័ន្ធទិន្នន័យ និងវិធីសាស្ត្រ (Data structure and Algorithms) គឺមានទំនាក់ទំនងគ្នាយ៉ាងជិតស្និទ្ធជាមួយគ្នាមានន័យថានៅពេលយើងនិយាយដល់បញ្ហាទិន្នន័យ (Data) ត្រូវគិតថាទិន្នន័យ (Data) នោះបានជះឥទ្ធិពលដល់វិធីសាស្ត្រណាមួយ (Algorithms) ណាមួយ ហើយនៅពេលគិតដល់បញ្ហាវិធីសាស្ត្រ (Algorithms) ត្រូវដឹងថាវិធីសាស្ត្រ (Algorithms) នោះបានជះឥទ្ធិពលលើទិន្នន័យ (Data) ណាដើម្បីនាំដល់លទ្ធផលចង់បាន។

សូមពិនិត្យមើលឧទាហរណ៍ដូចខាងក្រោម៖

ចូរលើឡើងពីតួនាទីរបស់រចនាសម្ព័ន្ធទិន្នន័យ និងវិធីសាស្ត្រ (Data structures and Algorithms) នៅក្នុងចំណោទបញ្ហា៖

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

Algorithms

$$\text{ពិភាក្សាលើមេគុណ } a \begin{cases} \text{បើ } a=0 \text{ នោះ } bx+c=0 \\ \text{បើ } a \neq 0 \text{ នោះ } ax^2+bx+c=0 \end{cases}$$

$$\text{សម្ព័ន្ធ } \Delta = b^2 - 4ac$$

ពិភាក្សាលើសញ្ញា Δ

- បើ $\Delta < 0$ នោះ No root
- បើ $\Delta = 0$ នោះ $X_1 = X_2 = -b / 2a$

$$\begin{aligned} &\bullet \text{ បើ } \Delta > 0 \text{ នោះ: } \begin{cases} X_1 = \frac{-b - \sqrt{\Delta}}{2a} \\ X_2 = \frac{-b + \sqrt{\Delta}}{2a} \end{cases} \\ &\text{រចនាសម្ព័ន្ធទិន្នន័យ (Data structure)} \end{aligned}$$

- ការរៀបចំរចនាកម្មវិធី
- ការប្រកាសអថេរ: a; b; c; X₁; X₂; delta
- បញ្ចូលអថេរចាំបាច់របស់ចំណោទ

៤. ភាសាសម្រាប់វិធីសាស្ត្រ

ពេលដោះស្រាយជាមួយម៉ាស៊ីន (Machine) យើងចាំបាច់ត្រូវគិតដល់វិធីសាស្ត្រ និងរចនាសម្ព័ន្ធ (Algorithms and structure) ហើយនៅពេលសម្តែងចេញរចនាសម្ព័ន្ធ (Structure) របស់វិធីសាស្ត្រ និងរចនាសម្ព័ន្ធទិន្នន័យ (Algorithms and Data structure) យើងត្រូវដាក់ចេញនូវភាសា (Language) របស់វិធីសាស្ត្រ និងរចនាសម្ព័ន្ធទិន្នន័យ (Algorithms and Data structure) របស់ចំណោទបញ្ជាឱ្យមានលក្ខណៈងាយស្រួលក្នុងការស្វែងយល់ និងងាយសរសេរ យើងគួរជ្រើសរើសនូវភាសាកម្មវិធីណាមួយ ដែលមានលក្ខណៈជាមូលដ្ឋាន ហើយមានគ្រប់លទ្ធភាពអាចធ្វើការឆ្លុះបញ្ចាំងរាល់រចនាសម្ព័ន្ធ (Structure) ដែលលើកឡើងដូចជា ប៉ាស្កាល់ (Pascal), ស៊ី/ស៊ីប៊ូកប៊ូក (C/C++) និងកម្មវិធីចាត់ (Java Programming) ។

ដូចនេះ ក្នុងឯកសារនេះ យើងជ្រើសរើសយកភាសាស៊ី/ស៊ីប៊ូកប៊ូក (C/C++) សម្រាប់ចេញនូវរាល់រចនាសម្ព័ន្ធ (structure) របស់វិធីសាស្ត្រ (Algorithms) ជាមូលដ្ឋានមួយចំនួនរបស់ភាសាកម្មវិធីស៊ី/ស៊ីប៊ូកប៊ូក (C/C++ Programming Language) ដូចខាងក្រោម៖

Header file

Global variable

Function

void main()

{

.....

.....

}

ចំពោះបកាសន៍ផ្ទេរ (Assignment Statement) ៖

V = E; (V=variable, E=Expression)

- អថេរ E ជាអ្នកផ្តល់តម្លៃឱ្យអថេរ V។
- អថេរ V ជា Variable, Object, ...។
- X = Y; X = 20; X = Y + 10; Y = X;

ចំពោះប្រភេទទិន្នន័យ (Data type)

គឺជាប្រភេទ Data ប្រើសម្រាប់ប្រកាសប្រាប់ពីលក្ខណៈអថេរ។

Data type រួមមាន ២ ប្រភេទគឺ៖

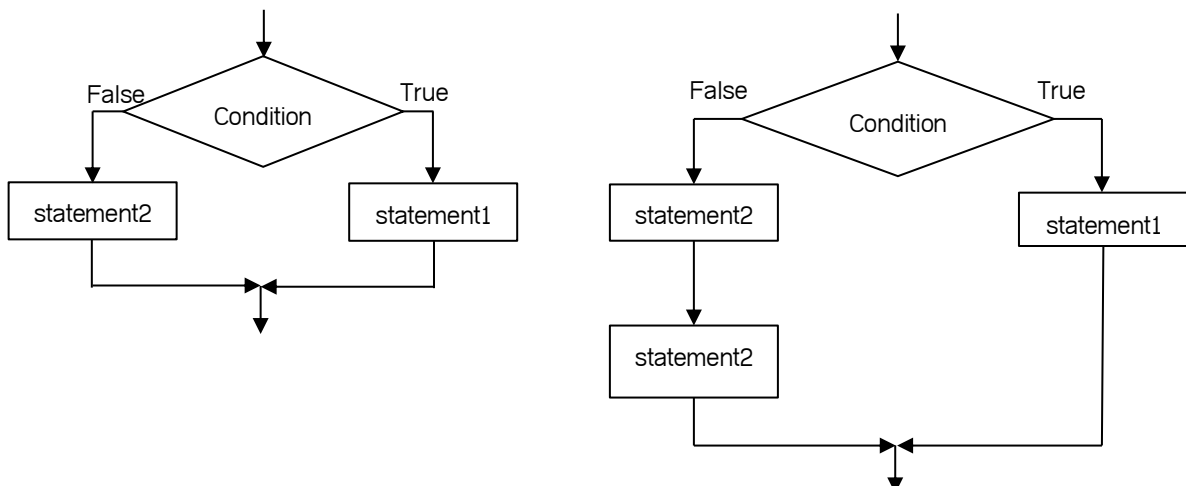
- (1) ប្រភេទទិន្នន័យធម្មតា (Simple data type): int, float, chr។
- (2) ប្រភេទរចនាសម្ព័ន្ធទិន្នន័យ (Structure data type): Array, String, Pointer, Enum, Struct, Union, and file។

ចំពោះបកាសន៍ (Statement)

1-បកាសន៍លក្ខខណ្ឌ (Control Statement)៖

```
if(condition){
    statements;
}
ឬ
if(condition){
    statement1;
}
else{
    statement2;statement3;
}
```

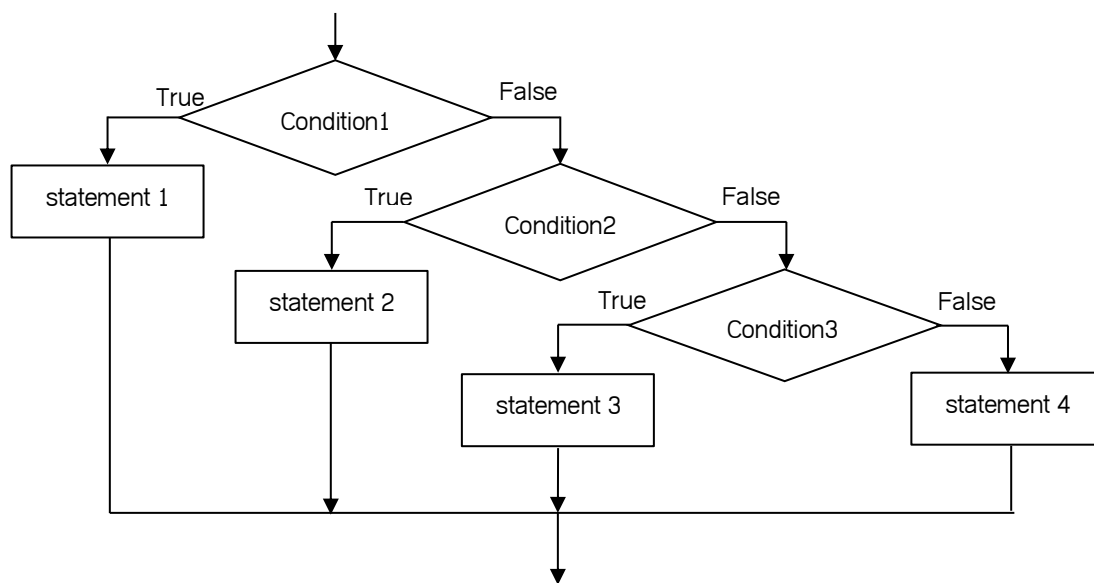
គំនូសតាងលំហូរ (Flow Chart)៖



```

if( Conditio1 ){
    Statement 1;
}
else if( condition 2 ){
    Statement 2;
}
.....
else{
    Statement( n );
}
    
```

គំនូសតាងលំហូរ (Flow Chart) ៖



បកាសន៍ Nested if() ៖

```

if ( condition ){
    statement 1;
}
else{
    if( condition ){
        statement 2;
    }
}
    
```

```

    }

    else{

        statement 3;

        statement 4;

        statement 5;

    }

}

```

បកាសន៍ Switch:

```

switch( expression ){

    Case const1:

statement1;

break;

    Case const2:

statement2;

break;

.....

    default: statement n;

}

```

2-បកាសន៍បង្វិលជុំ (Loop statement):

for Loop:

```

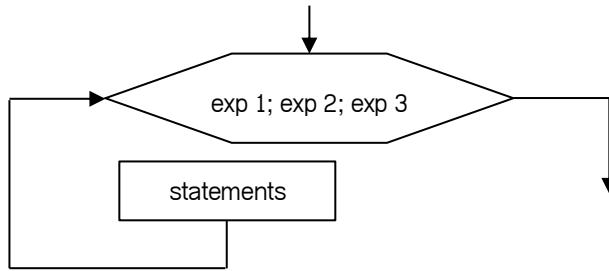
for( exp 1; exp 2; exp 3 ){

    statements;

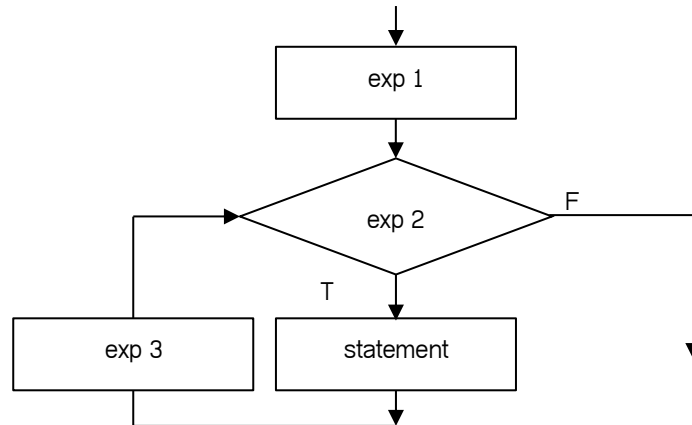
}

```

គំនូសតាងលំហូរ (Flow Chart)៖



ឬ



បកស្រង់បង្វិលជុំ while (while Loop statements) ៖

expression 1;

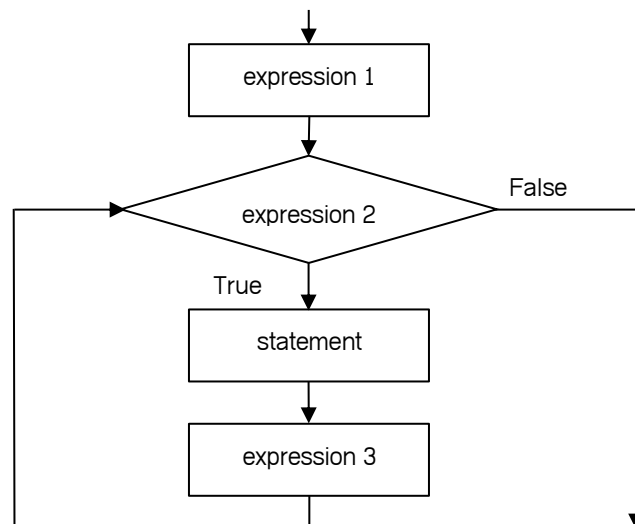
while(expression 2){

statement;

expression 3;

}

គំនូសតាងលំហូរ (Flow Chart)៖



បកស្រង់បង្វិលជុំ do/while (do/while loop Statement) ៖

exp 1;

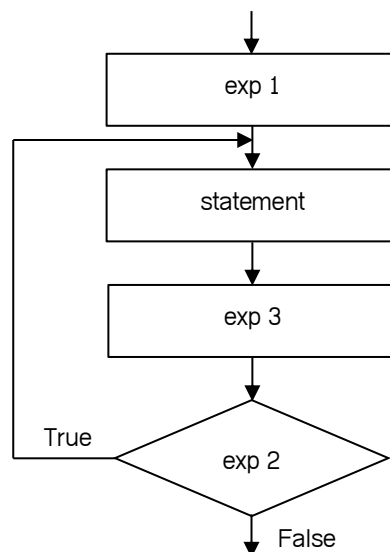
do {

statement;

exp3;

}while(exp2);

គំនូសតាងលំហូរ (Flow Chart) ៖



ឧទាហរណ៍១៖

ចូរសរសេរកម្មវិធី (program) ដើម្បីត្រួតពិនិត្យមើលតួអក្សរដែលវាយបញ្ចូល yes ឬ no។

ចំពោះភាសា C Programming ៖

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

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

```
main( )
```

```
{
```

```
char ch;
```

```
printf( "Enter your choice y/n:" );scanf( "%c",&ch );
```

```
if( ch=='Y' || ch=='y' )
```

```
printf( "Your choice is yes\n" );
```

```

else

    printf("Your choice is no\n");

getch( );

return 0;

}

```

សូមចុចលើ Ctrl+F9 ដើម្បីពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំពោះភាសា C++ Programming

```

#include<iostream.h>

#include<conio.h>

main( )

{

    char ch;

    cout<<"Enter your choice y/n:">>cin>>ch;

    if( ch=='Y' || ch=='y' )

        cout<<"Your choice is yes\n";

    else

        cout<<"Your choice is no\n";

    getch( );

    return 0;

}

```

សូមចុចលើ Ctrl+F9 ដើម្បីពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ឧទាហរណ៍២៖

សរសេរកម្មវិធីដើម្បីត្រួតពិនិត្យចំនួនបឋមរឺមិនបឋម

ចំពោះភាសា C Programming៖

```
#include<stdio.h>

#include<conio.h>

main( )
{
    /*Program to determine whether a number is Prime or not*/
    clrscr( );
    int num,i;
    printf("Enter a number:");scanf( "%d",&num );
    i=2;
    while( i<=num-1 )
    {
        if( num %i==0 )
        {
            printf("Number is not a prime");
            break;
        }
    }
}
```

```

    }

    i=i+1;

}

if( i==num)

    printf( "Prime number %d",num );

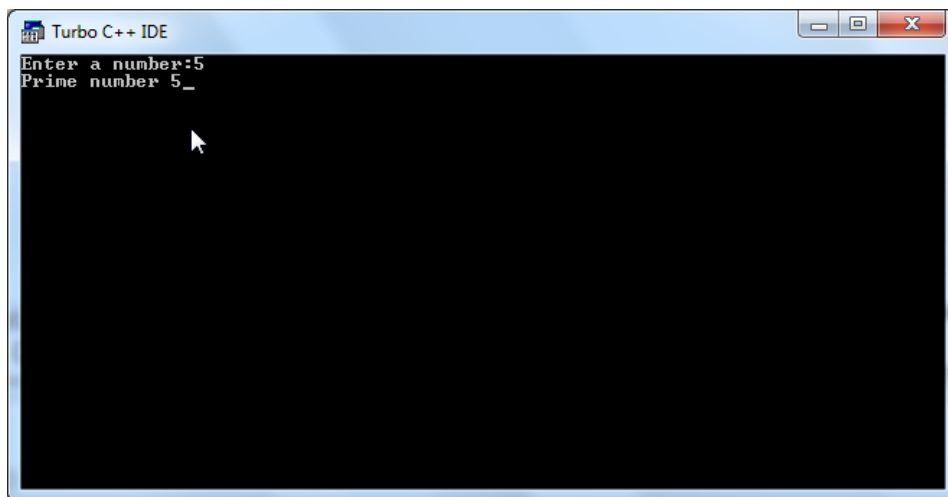
getch( );

return 0;

}

```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំពោះភាសា C++ Programming៖

```

#include<iostream.h>

#include<conio.h>

main( )

{

    /*Program to determine whether a number is Prime or not*/

    clrscr( );

    int num,i;

    cout<<"Enter a number:";cin>>num;

    i=2;

```

```
while( i<=num-1 )
{
    if( num %i==0 )
    {
        cout<<"Number is not a prime";
        break;
    }
    i=i+1;
}
```

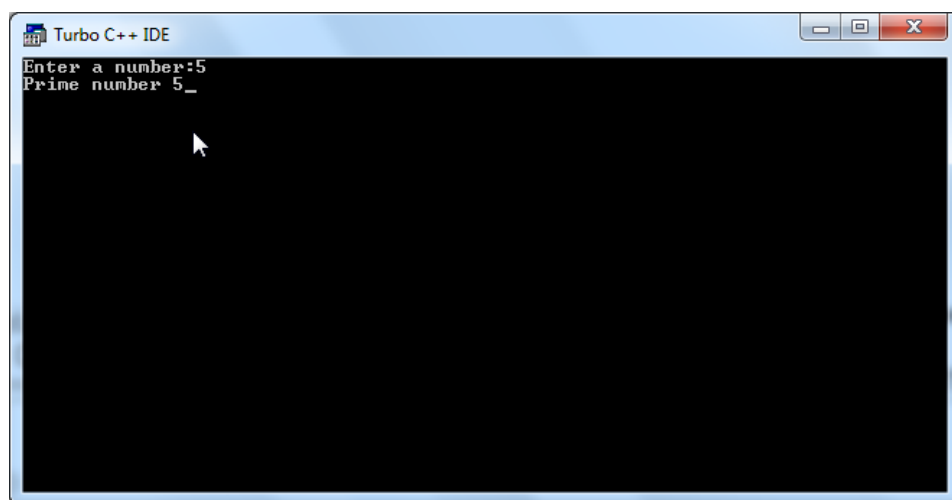
```
if( i==num )
    cout<<"Prime number "<<num;
```

```
getch( );
```

```
return 0;
```

```
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ឧទាហរណ៍៣៖ សរសេរកម្មវិធីដើម្បីគណនាផលបូក ផលដកនិងផលគុណដោយប្រើប្រាស់បកាសន៍
(Statement) Switch

ចំពោះភាសា C Programming

```
#include<stdio.h>

#include<conio.h>

main( )
{
    clrscr( );

    /*Demonstration of switch statement */

    int a,b, sum, diff, pro, ch;

    float rat;

    printf( "Enter value of a and b:\n" );

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

    printf( "\n M E N U" );

    printf( "\n 1 for Sum" );

    printf( "\n 2 for Product" );

    printf( "\n 3 for Ratio" );

    printf( "\n 4 for Difference" );

    printf( "\n Enter your choice 1/ 2/ 3/ 4:" );

    scanf( "%d",&ch );

    switch( ch ){

        case 1:

            sum =a+b;

            printf( "sum of a & b=%d",sum );
```

```

        break;

    case 2:

        pro=a *b;

        printf( "Product of a & b=%d",pro );

        break;

    case 3:

        rat=a/b;

        printf( "Ratio of a & b=%f",rat );

        break;

    case 4:

        diff=a-b;

        printf( "Diff of a & b=%d",diff );

        break;

}

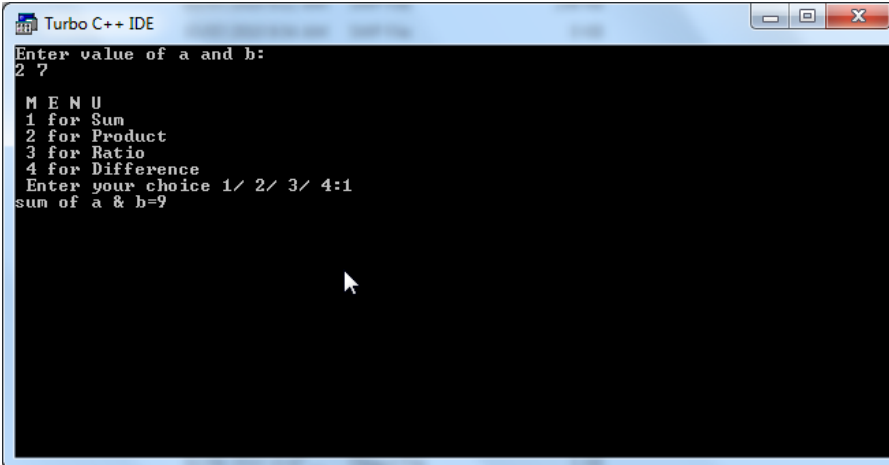
getch( );

return 0;

}

```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



The screenshot shows the Turbo C++ IDE window. The output window displays the following text:

```

Enter value of a and b:
2 7

M E N U
1 for Sum
2 for Product
3 for Ratio
4 for Difference
Enter your choice 1/ 2/ 3/ 4:1
sum of a & b=9

```

ចំពោះភាសា C++ Programming៖

```

#include<iostream.h>

#include<conio.h>

main( )
{
    clrscr( );

    /*Demonstration of switch statement */

    int a,b, sum, diff, pro, ch;

    float rat;

    cout<<"Enter value of a and b:\n";

    cin>>a>>b;


    cout<<"\n M E N U";

    cout<<"\n 1 for Sum";

    cout<<"\n 2 for Product";

    cout<<"\n 3 for Ratio";

    cout<<"\n 4 for Difference";

    cout<<"\n Enter your choice 1/ 2/ 3/ 4:";

    cin>>ch;


    switch( ch){

        case 1:

            sum =a+b;

            cout<<"sum of a & b= "<<sum;

            break;

```


case 2:

pro=a *b;

cout<<"Product of a & b = "<<pro;

break;

case 3:

rat=a/b;

cout<<"Ratio of a & b= "<<rat;

break;

case 4:

diff=a-b;

cout<<"Diff of a & b= "<<diff;

break;

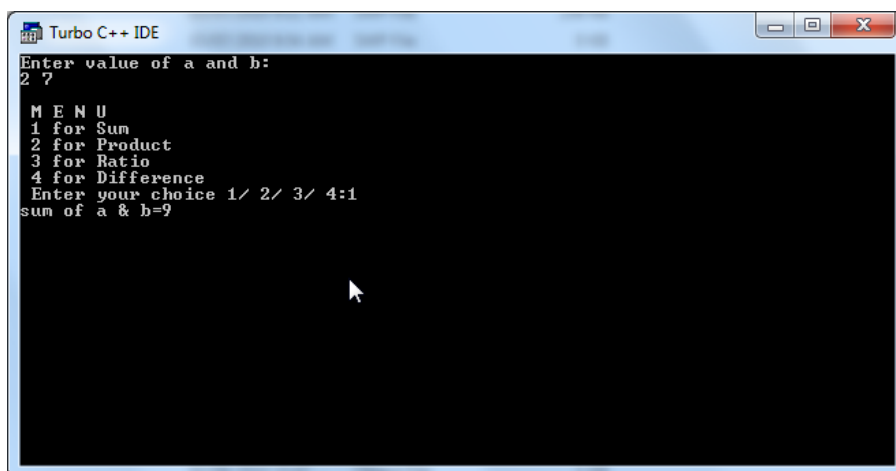
}

getch();

return 0;

}

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



Structure of Function

Return_type function_name(list parameter.....)

```
{
    Local variable
    statement;
    .....
}
```

ក្នុងនោះ return type រួមមាន ២ ប្រភេទគឺ void & datatype។

- function_name ជាឈ្មោះរបស់ function។
- List parameter រួមមាន ២ប្រភេទ៖

Value parameter គឺជាប៉ារ៉ាម៉ែត្រ (Parameter) ដែលមិនប្រែប្រួលតម្លៃមុនពេល និងក្រោយពេលហៅអនុគមន៍ (call function)។

Reference parameter គឺជា parameter ប្រែប្រួលតម្លៃមុនពេលហៅ និងក្រោយពេលហៅអនុគមន៍ (call function)។

ឧទាហរណ៍៖

អនុគមន៍ប្តូរតម្លៃ នៃ ២ចំនួនគត់ (ដោយប្រើលក្ខណៈ Value parameter និង reference parameter)

ចំពោះ Value parameter៖

ចំពោះភាសា C Programming

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

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

```
void change( int a, int b);
```

```
main( )
```

```
{
    clrscr( );
```

```

int x,y;

printf( "Input a value: " );scanf( "%d",&x);

printf( "Input b value: " );scanf( "%d",&y );

change( x,y );

printf( "\nafter call function change\n" );

printf( "x=%d\n",x );

printf( "y=%d",y );

getch( );

return 0;

}

void change( int a, int b )

{

    int temp=a;

    a=b;

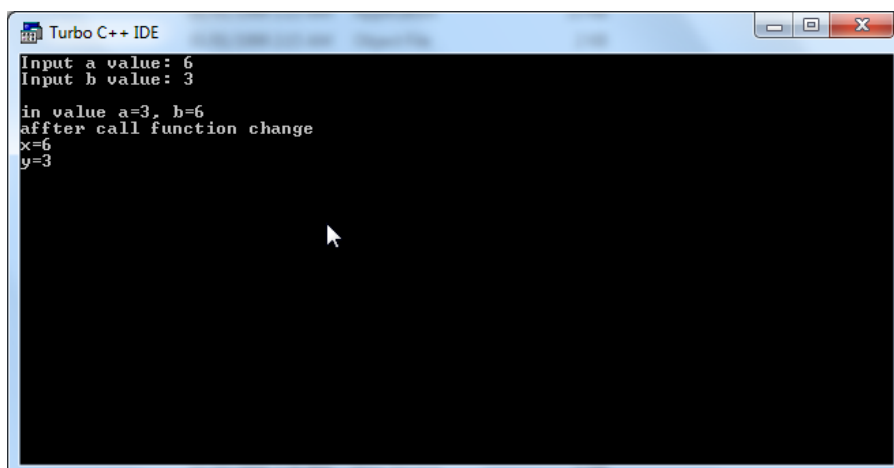
    b=temp;

    printf( "\nin value a=%d, b=%d", a, b );

}

```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



```

Turbo C++ IDE
Input a value: 6
Input b value: 3
in value a=3, b=6
after call function change
x=6
y=3

```

ចំពោះភាសា C++ Programming៖

```
#include<iostream.h>
```

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

```
void change( int a, int b);
```

```
main( )
```

```
{
```

```
clrscr( );
```

```
int x,y;
```

```
cout<<"Input a value: "; cin>>x;
```

```
cout<<"Input b value: "; cin>>y;
```

```
change( x,y );
```

```
cout<<endl<<"affter call function change"<<endl;
```

```
cout<<"x= "<<x<<endl;
```

```
cout<<"y= "<<y<<endl;
```

```
getch( );
```

```
return 0;
```

```
}
```

```
void change( int a, int b )
```

```
{
```

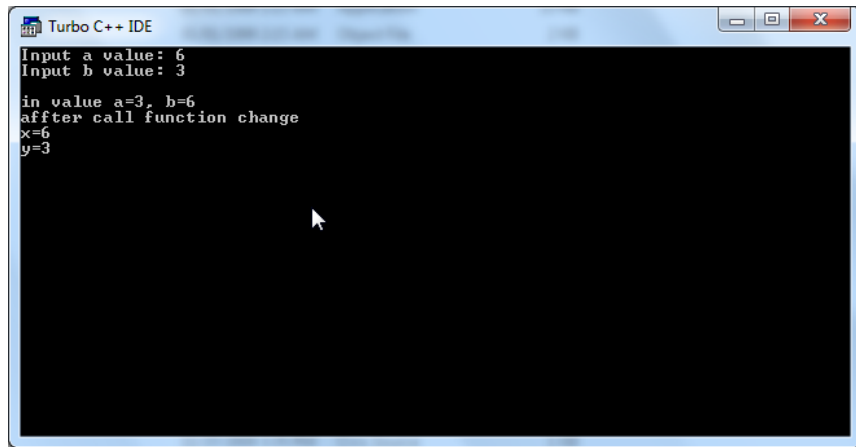
```
int temp=a;
```

```
a=b;
```

```
b=temp;
```

```
cout<<endl<<"in value a="<<a<<" b= "<<b;
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



Reference parameter

ចំពោះភាសា C Programming

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

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

```
void change(int *a, int *b);
```

```
main( )
```

```
{
```

```
clrscr( );
```

```
int x,y;
```

```
printf("Input a value: ");
```

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

```
printf("Input b value: ");
```

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

```
change( &x, &y );
```

```
printf("\nafter call function change\n");
```

```
printf("x=%d\n",x);

printf("y=%d",y);

getch( );

return 0;

}
```

```
void change( int *a, int *b)

{

    int temp= *a;

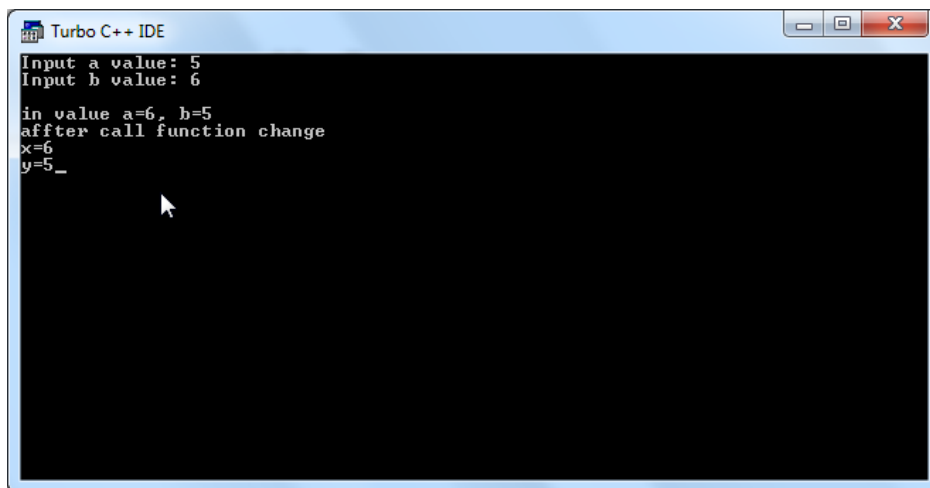
    *a= *b;

    *b=temp;

    printf("\nin value a=%d, b=%d", *a, *b);

}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



```
Turbo C++ IDE
Input a value: 5
Input b value: 6
in value a=6, b=5
affter call function change
x=6
y=5_
```

ចំពោះភាសា C++ Programming

```
#include<iostream.h>
```

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

```
void change( int *a, int *b);
```

```

main( )
{
    clrscr( );
    int x,y;
    cout<<"Input a value: "; cin>>x;
    cout<<"Input b value: "; cin>>y;

    change( &x, &y );

    cout<<endl<<"affter call function change"<<endl;
    cout<<"x= "<<x<<endl;
    cout<<"y= "<<y<<endl;
    getch( );
    return 0;
}

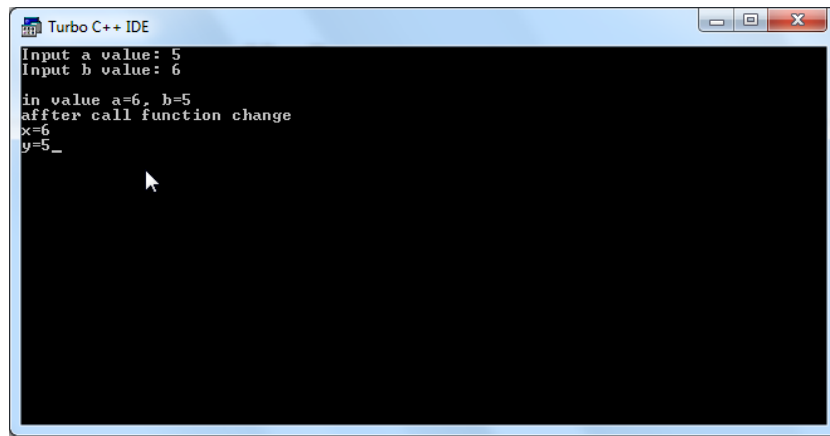
```

```

void change( int *a, int *b )
{
    int temp= *a;
    *a= *b;
    *b=temp;
    cout<<endl<<"in value a="<< *a<<" b= "<< *b;
}

```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



```

Turbo C++ IDE
Input a value: 5
Input b value: 6
in value a=6, b=5
after call function change
x=6
y=5_
    
```

យើងអាចបង្កើត Reference ម្យ៉ាងទៀតបានដូចខាងក្រោម:

ចំពោះភាសា C Programming:

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

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

```
void change( int &a, int &b );
```

```
main( )
```

```
{
```

```
clrscr( );
```

```
int x,y;
```

```
printf( "Input a value: ");scanf( "%d",&x);
```

```
printf( "Input b value: ");scanf( "%d",&y);
```

```
change( &x, &y);
```

```
printf( "\nafter call function change\n");
```

```
printf( "x=%d\n",x);
```

```
printf( "y=%d",y);
```

```
getch( );
```

```
return 0;
```


}

```
void change(int *a, int *b)
```

```
{
```

```
    int temp= *a;
```

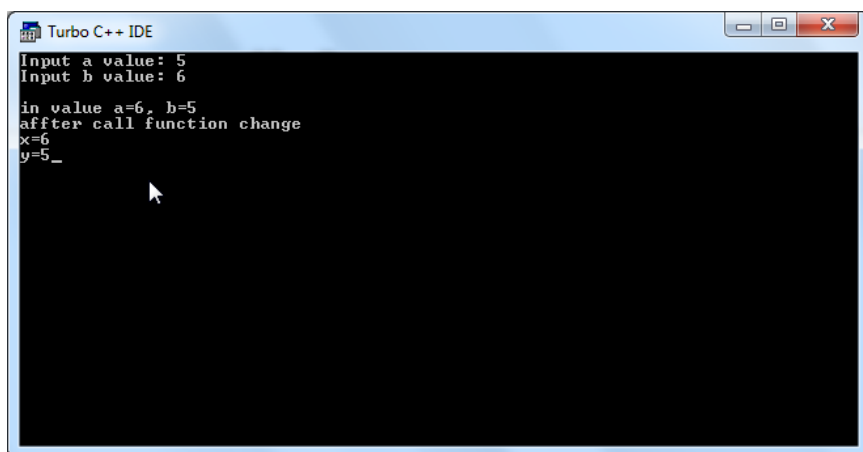
```
    *a= *b;
```

```
    *b=temp;
```

```
    printf("\nin value a=%d, b=%d", *a, *b);
```

```
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំពោះភាសា C++ Programming៖

```
#include<iostream.h>
```

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

```
void change(int &a, int &b);
```

```
main( )
```

```
{
```

```
    clrscr( );
```

```
    int x,y;
```

```
cout<<"Input a value: "; cin>>x;
```

```
cout<<"Input b value: "; cin>>y;
```

```
change(x, y);
```

```
cout<<endl<<"affter call function change"<<endl;
```

```
cout<<"x= "<<x<<endl;
```

```
cout<<"y= "<<y<<endl;
```

```
getch( );
```

```
return 0;
```

```
}
```

```
void change( int &a, int &b )
```

```
{
```

```
int temp=a;
```

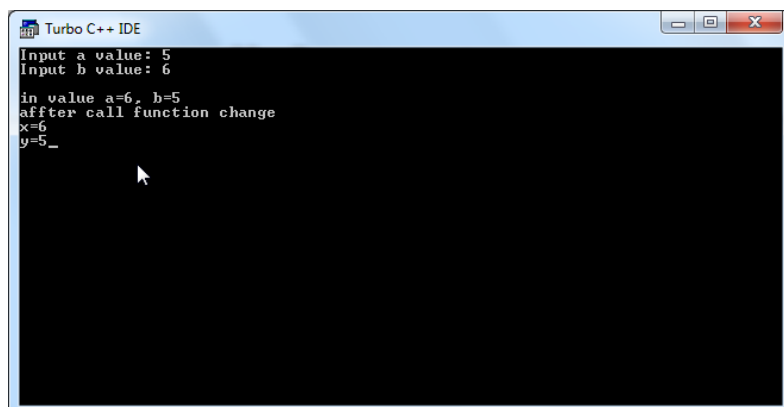
```
a=b;
```

```
b=temp;
```

```
cout<<endl<<"in value a="<<a<<" b= "<<b;
```

```
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



```
Turbo C++ IDE
Input a value: 5
Input b value: 6
in value a=6, b=5
affter call function change
x=6
y=5_
```

ចំណាំ៖ យើងមានរបៀបសរសេរអនុគមន៍ ៤ បែប ដែលត្រូវបានលើកឡើងគឺ៖

១-អនុគមន៍មានប្រភេទទិន្នន័យ (Data Type) និងមានប៉ារ៉ាម៉ែត្រ (Parameter)

ឧទាហរណ៍៖ គណនាផលបូក ២ចំនួនគត់ a និង b

```
int sum1( int a, int b){
    int S=a+b;
    return( s);
}
```

ឧទាហរណ៍៖ Value parameter

void change(int x, int y); /* x and y ជា value parameter */

ឧទាហរណ៍៖ Reference parameter

void change(int *x, int *y); /* x and y ជា Return parameter */

ឧទាហរណ៍៖

void change(int &x, int &y); /* x and y គឺជា reference parameter */

២-អនុគមន៍គ្មានប្រភេទទិន្នន័យ (Data Type) និងគ្មានប៉ារ៉ាម៉ែត្រ (Parameter)

```
int sum2( ){
    int a, b;
    printf( "input a, b:" );
    //cout<<"input a, b:";
    //System.out.printf( "input a, b:" );
    scanf( "%d%d",&a, &b );
    //cin>>a>>b;
    //Scanner kbd = new Scanner( System.in );
    //a = kbd.nextInt( );
    //b = kbd.nextInt( );
    int s=a+b return( s);
}
```

៣-អនុគមន៍គ្មាន data type និងគ្មាន parameter

```
void sum3(int a, int b){
    int s =a+b;
    printf( "\n sum =%d",s);
    //cout<<endl<<"sum ="<<s;
    //System.out.println( "sum =" + s );
}
```

៤-អនុគមន៍គ្មាន data type និងគ្មាន parameter

```
void sum( ){
    int a, int b;
    printf( "input a, b:" );
    //cout<<"input a, b:";
    //System.out.printf( "input a, b:" );

    scanf( "%d%d", &a, &b );
    //cin>>a>>b;
    //Scanner kbd = new Scanner( System.in );
    //a = kbd.nextInt( );
    //b = kbd.nextInt( );

    int s=a+b;
    printf( "\n sum =%d",s);
    //cout<<endl<<"sum ="<<s;
    //System.out.println( "sum =" );
}
```

ឧទាហរណ៍៖ ចូរសរសេរកម្មវិធីដែលត្រូវបានប្រើប្រាស់ called ដោយតម្លៃ value parameter passing method។ ដើម្បីធ្វើផលបូក នៃពីរចំនួន។

ចំពោះភាសា C Programming៖

```
#include<stdio.h>

#include<conio.h>

int sumTwoValue(int x, int y);

main( )
{
    clrscr( );

    int a, b;

    printf("input a,b:");

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


    int sum;

    sum = sumTwoValue(a,b);

    printf("Sum of a and number is %d",sum );

    getch( );

    return 0;
}

int sumTwoValue(int x, int y)
{
    return (x+y);
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំពោះភាសា C++ Programming៖

```
#include<iostream.h>

#include<conio.h>

int sumTwoValue(int x, int y);

main( )
{
    clrscr( );

    int a, b;

    cout<<"input a,b:";

    cin>>a>>b;

    int sum;

    sum = sumTwoValue( a,b);

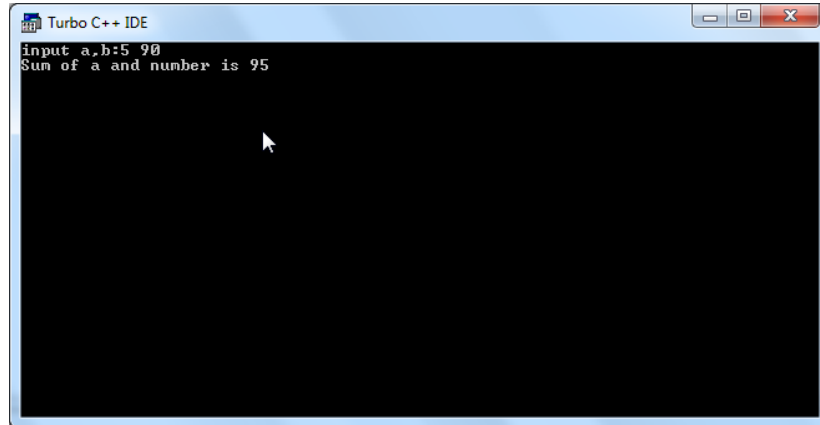
    cout<<"Sum of a and number is"<<sum;

    getch( );

    return 0;
}

int sumTwoValue( int x, int y)
{
    return (x+y);
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ឧទាហរណ៍២៖ សរសេរកម្មវិធីដោយប្រើអនុគមន៍ call by reference mechanism។

ចំពោះភាសា C Programming

```
#include<stdio.h>

#include<conio.h>

main( )
{
    clrscr( );

    int a, b;

    void function( int *, int *y);

    a =30;

    b=50;

    printf( "a=%d, b=%d before function call \n",a,b);

    function( &a, &b);

    printf( "a=%d, b=%d after function call\n",a,b);

    getch( );

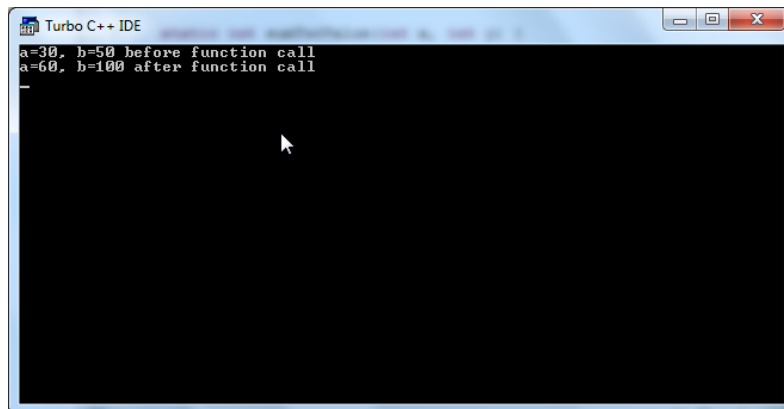
    return 0;

}

/* Call by reference function */
```

```
void function(int *x, int *y)
{
    *x= *x+ *x;
    *y= *y+ *y;
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំពោះភាសា C++ Programming៖

```
#include<iostream.h>
#include<conio.h>
main( )
{
    clrscr( );
    int a, b;
    void function(int *, int *y);
    a =30;
    b=50;
    cout<<"a="<<a<<" , "<<b<<" before function call."<<endl;
    function( &a, &b);
    cout<<"a="<<a<<" , "<<b<<" after function call."<<endl;
```



```

    getch( );

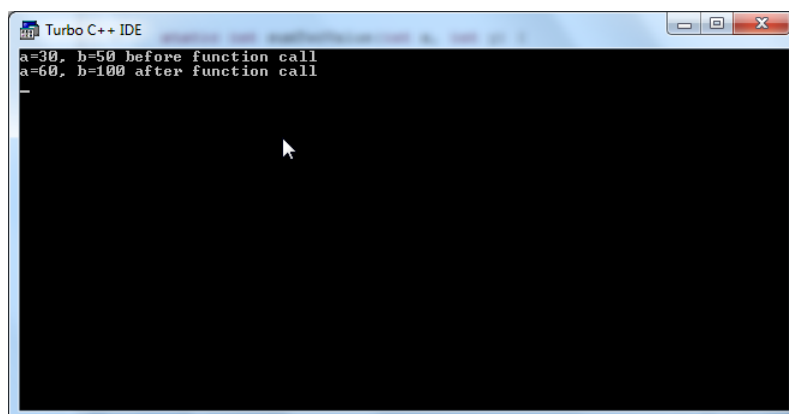
    return 0;

}

/* Call by reference function */
void function(int *x, int *y)
{
    *x= *x+ *x;
    *y= *y+ *y;
}

```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំណាំ យើងអាចប្រើប្រាស់ Automatic Storage Class។ សម្រាប់កំណត់ local variable ឱ្យមាន local lifetime។

សូមពិនិត្យមើលទម្រង់របស់ វាដូចខាងក្រោម៖

```

main( )
{
    auto int a, b, c;

    .....

    .....

}

```

ឧទាហរណ៍១៖

ចូរសរសេរកម្មវិធី (Program) ដើម្បីបង្ហាញពីរបៀបប្រើប្រាស់ automatic variable។

ចំពោះភាសា C Programming៖

```
#include<stdio.h>

#include<conio.h>

void functionx( );

void functiony( );

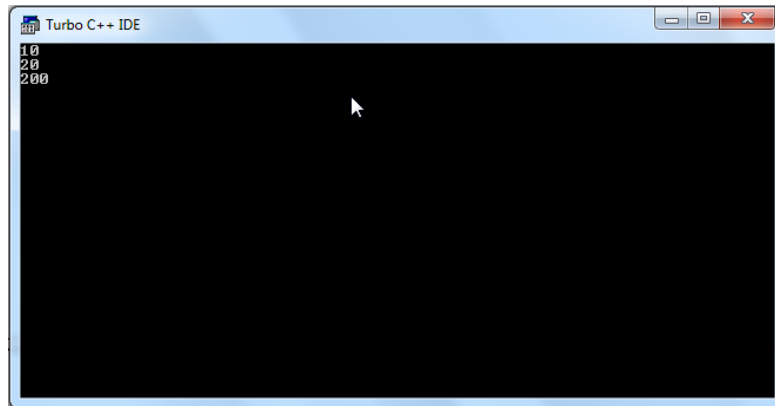
main( )
{
    clrscr( );
    auto int num=200;
    functionx( );
    printf( "%d\n", num );
    getch( );
    return 0;
}

void functionx( )
{
    auto int num=20;
    functiony( );
    printf( "%d\n",num );
}

void functiony( )
{
    auto int num=10;
```

```
printf( "%d\n",num );
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំពោះភាសា C++ Programming៖

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

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

```
void functionx( );
```

```
void functiony( );
```

```
main( )
```

```
{
```

```
clrscr( );
```

```
auto int num=200;
```

```
functionx( );
```

```
cout<<num<<endl;
```

```
getch( );
```

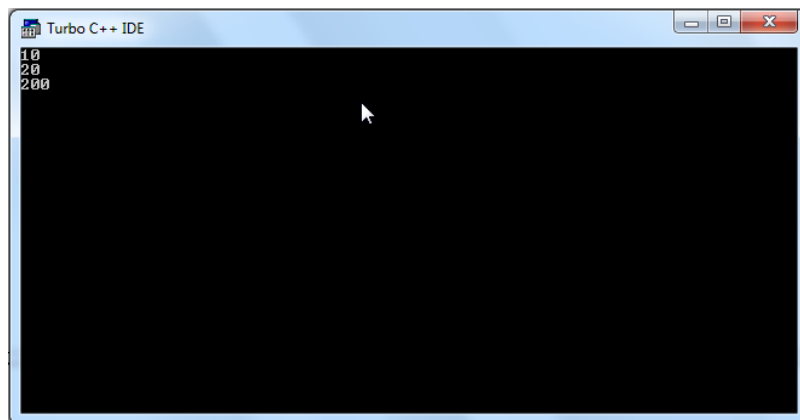
```
return 0;
```

```
}
```

```
void functionx( )
{
    auto int num=20;
    functiony( );
    cout<<num<<endl;
}
```

```
void functiony( )
{
    auto int num=10;
    cout<<num<<endl;
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ឧទាហរណ៍២៖

សរសេរកម្មវិធីដើម្បីបំប្លែងពីប្រព័ន្ធគោលដប់ទៅ ប្រព័ន្ធគោលដប់ពីរ ដោយប្រើអនុគមន៍ (function) ។

ចំពោះភាសា C Programming៖

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

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

```

int dec_to_bin( int b );

main( )
{
    auto int dec, bin;

    int dec_to_bin( int );

    printf( "Enter a decimal number : " );

    scanf( "%d",&dec );

    bin=dec_to_bin( dec );

    printf( "The decimal number is =%d\n",dec );

    printf( "The binary number is=%d\n",bin );


    getch( );

    return 0;
}

/*Function to find the binary equivalent*/
int dec_to_bin( int d )
{
    auto int b, r, y;

    b=0;

    y=1;

    while( d>0 )
    {
        r = d%2;

        d = d/2;

```

```

        b = b+(r*y);

        y = y*10;

    }

```

```

return(b);

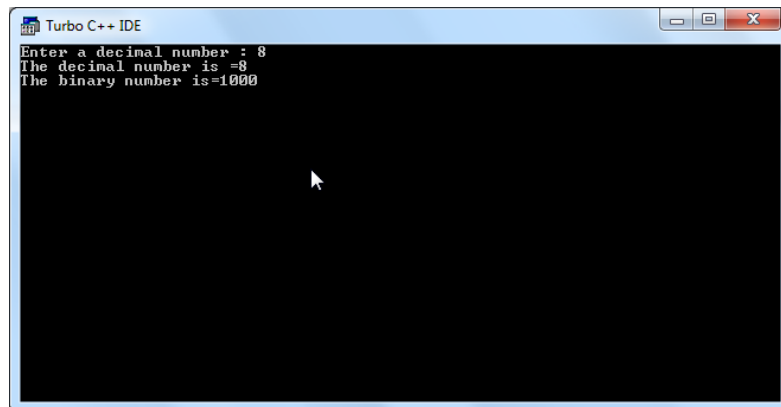
```

```

}

```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំពោះភាសា C++ Programming៖

```

#include<iostream.h>

```

```

#include<conio.h>

```

```

int dec_to_bin( int b );

```

```

main( )

```

```

{

```

```

    clrscr( );

```

```

    auto int dec, bin;

```

```

    int dec_to_bin( int );

```

```

    cout<<"Enter a decimal number : ";

```

```

    cin>>dec;

```

```

    bin=dec_to_bin( dec );

```

```

        cout<<"The decimal number is = "<<dec<<endl;

        cout<<"The binary number is= "<<bin<<endl;


        getch( );

        return 0;

}

```

/*Function to find the binary equivalent*/

```

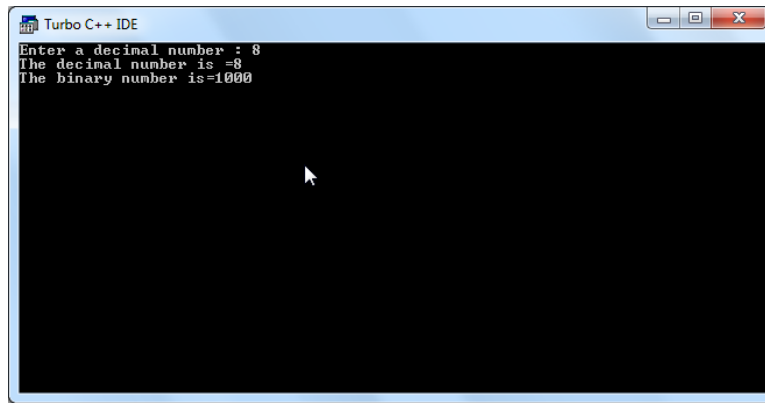
int dec_to_bin( int d)
{
    auto int b, r, y;

    b=0;
    y=1;
    while( d>0 )
    {
        r = d%2;
        d = d/2;
        b = b+(r*y);
        y = y*10;
    }

    return( b);
}

```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



Keyword **extern** បង្ហាញថា actual storage និង initail value នៃ variable, ឬ body នៃ function។

extern <data definition>;

entern <function prototype>;

ឧទាហរណ៍១៖

ចូរសរសេរសរកម្មវិធី (Program) ដើម្បីកំណត់ properties នៃ external variables។

ចំពោះភាសា C Programming

សូមបង្កើត file header មួយឈ្មោះ TESTEXTE.H

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

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

```
int num;
```

```
void output( )
```

```
{
```

```
    printf( "%d", num );
```

```
}
```

សូមបង្កើត file header មួយឈ្មោះ TESTEXTE.H

```
#include<TestExte.h>
```



```
main( )
{
    extern int num;

    clrscr( );

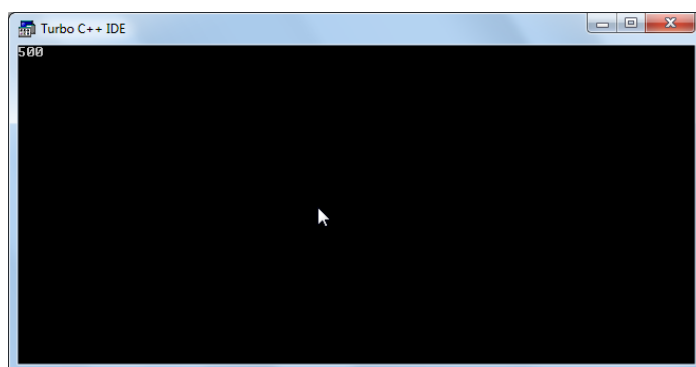
    num=500;

    output( );

    getch( );

    return 0;
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំពោះភាសា C++ Programming

សូមបង្កើត file header មួយឈ្មោះ TESTEXTE.H

```
#include<iostream.h>

#include<conio.h>

int num;

void output( )
{
    cout<<num;
}
```

សូមបង្កើត file header មួយឈ្មោះ TESTEXTE.H

```
#include<TestExte.h>
```

```
main( )
```

```
{
```

```
    extern int num;
```

```
    clrscr( );
```

```
    num=500;
```

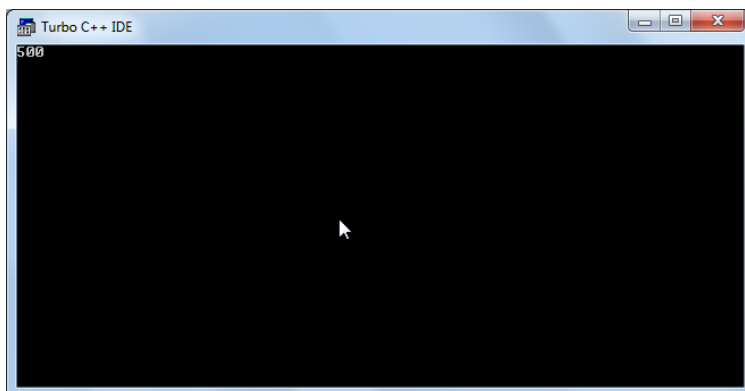
```
    output( );
```

```
    getch( );
```

```
    return 0;
```

```
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ឧទាហរណ៍២៖

សរសេរកម្មវិធីដើម្បីត្រឡប់ម៉ាទ្រីសដោយប្រើ extern variables។

ចំពោះភាសា C Programming

សូមបង្កើត header file DSEXTERN.H ដែលមាន កូដដូចខាងក្រោម៖

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

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

```
int a[10][10],b[10][10];
```

```
int i, j, n;
```

```
/* Function to transpose a matrix */
```

```
void transpose( )
```

```
{
```

```
    for( i=0;i<n;i++ )
```

```
        for( j=0;j<n;j++ )
```

```
            b[i][j]=a[i][j];
```

```
}
```

```
/* Function to print a matrix */
```

```
void matprint( int x[10][10] )
```

```
{
```

```
    for( i=0;i<n;i++ )
```

```
    {
```

```
        for( j=0;j<n;j++ )
```

```
            printf( "\t%d", x[i][j] );
```

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

```
    }
```

```
}
```

សូមបង្កើត file DS_MainExternal.CPP ដែល មានកូដដូចខាងក្រោម៖

```
#include<dsextern.h>
```

```
#include<stdlib.h>
```

```
main( )
```

```
{
    extern int a[10][10],b[10][10];

    extern int i, j, n;

    clrscr( );

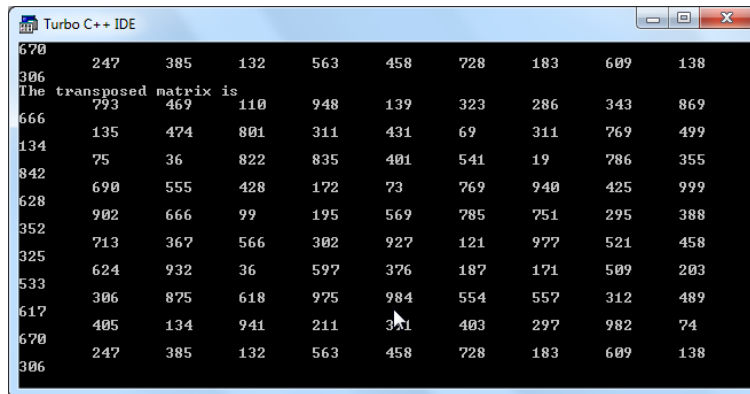
    printf( "Enter order of the matrix\n" );
    scanf( "%d",&n );
    printf( "\n Enter matrix elements\n" );


    randomize( );
    for( i=0; i<n; i++ )
        for( j=0; j<n;j++ )
            a[i][j]=random( 1000 );


    printf( "The original matrix is \n" );
    matprint( a );
    transpose( );
    printf( "The transposed matrix is \n" );
    matprint( b );


    getch( );
    return 0;
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំពោះភាសា C++ Programming

សូមបង្កើត header file DSEXTERN.H ដែលមាន កូដដូចខាងក្រោម៖

```
#include<iostream.h>
```

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

```
int a[10][10],b[10][10];
```

```
int i, j, n;
```

```
/* Function to transpose a matrix */
```

```
void transpose( )
```

```
{
```

```
    for( i=0;i<n;i++)
```

```
        for( j=0;j<n;j++)
```

```
            b[i][j]=a[j][i];
```

```
}
```

```
/* Function to print a matrix */
```

```
void matprint( int x[10][10] )
```

```
{
```

```
    for( i=0;i<n;i++)
```

```
{
    for( j=0;j<n;j++ )
        cout<<"\t"<<x[i][j] );
    cout<<"\n";
}
}
```

សូមបង្កើត file DS_MainExternal.CPP ដែល មានកូដដូចខាងក្រោម៖

```
#include<dsextern.h>
```

```
#include<stdlib.h>
```

```
main( )
```

```
{
    extern int a[10][10],b[10][10];
    extern int i, j, n;

    clrscr( );

    printf( "Enter order of the matrix\n" );
    scanf( "%d",&n );
    printf( "\n Enter matrix elements\n" );
    randomize( );
    for( i=0; i<n; i++ )
        for( j=0; j<n;j++ )
            a[i][j]=random( 1000 );

    printf( "The original matrix is \n" );
    matprint( a );
    transpose( );
```

```
printf( "The transposed matrix is \n" );

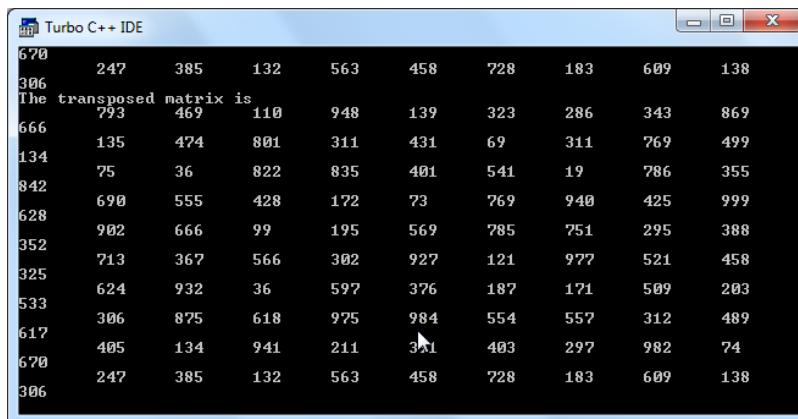
matprint( b );

getch( );

return 0;

}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



```
Turbo C++ IDE
670      247      385      132      563      458      728      183      609      138
306  The transposed matrix is
666      793      469      110      948      139      323      286      343      869
134      135      474      801      311      431      69      311      769      499
842      75      36      822      835      401      541      19      786      355
628      690      555      428      172      73      769      940      425      999
352      902      666      99      195      569      785      751      295      388
325      713      367      566      302      927      121      977      521      458
533      624      932      36      597      376      187      171      509      203
617      306      875      618      975      984      554      557      312      489
670      405      134      941      211      351      403      297      982      74
306      247      385      132      563      458      728      183      609      138
```

យើងអាចប្រើប្រាស់ keyword static សម្រាប់ឱ្យតម្លៃរបស់ variable ចងចាំរហូតដល់កម្មវិធីបិទ។

```
static <variable>;
```

```
static <functions definition>;
```

ឧទាហរណ៍១៖

ចូរសរសេរកម្មវិធី (Program) ដើម្បីប្រើប្រាស់ static variable។

ចំពោះភាសា C Programming៖

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

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

```
void staticfun( );
```

```
main( )
```

```
{
```

```

clrscr( );

int n;

for( n=1;n<=3;n++ )

    staticfun( );


getch( );

return 0;

}

void staticfun( )

{

    static int s=0; /* s value assigned to 0 at the time of compilation */

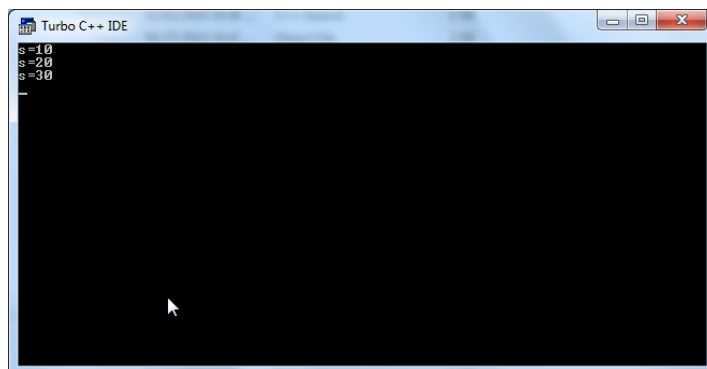
    s=s+10;

    printf( "s=%d\n",s);

}

```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំពោះភាសា C++ Programming៖

```
#include<iostream.h>
```

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

```
void staticfun( );
```



```
main( )
{
    clrscr( );

    int n;

    for( n=1;n<=3;n++ )
        staticfun( );

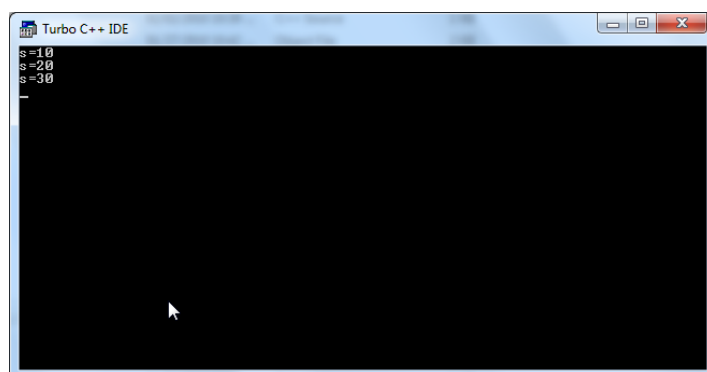
    getch( );
    return 0;
}

void staticfun( )
{
    static int s=0; /* s value assigned to 0 at the time of compilation */

    s=s+10;

    cout<<"s="<<s<<endl;
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ឧទាហរណ៍២៖

សូមសរសេរកម្មវិធីដើម្បីកំណត់ Fibonacci ដោយប្រើប្រាស់ static variable។

ចំពោះភាសា C Programming៖

```
#include<stdio.h>

#include<conio.h>

main( )
{
    clrscr( );

    int i,n;

    int fibo( int );

    printf( "Enter number of elements in the series \n" );

    scanf( "%d",&n );

    printf( "\n Fibonacci number \n\n" );

    for( i=0;i<n;i++)

        printf( "%d\t",fibo( i ) );

    getch( );

    return 0;
}

/* Function to find Fibonacci number */

int fibo( int k )
{
    static int n1=1;

    static int n2=1;

    int n3;
```

```

        if( k==1 )
            n3=0;

        else if( k==2 )
            n3=1;

        else
            n3=n1+n2;

        n1=n2;
        n2=n3;
        return( n3 );
    }

```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំពោះភាសា C++ Programming៖

```
#include<iostream.h>
```

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

```
int fibo( int k );
```

```
main( )
```

```
{
```

```
    clrscr( );
```

```
    int i,n;
```

```

int fibo( int );

cout<<"Enter number of elements in the series<<"<<endl;

cin>>n;

cout<<endl<<"Fibonacci number "<<endl<<endl;

for( i=0;i<n;i++ )

    cout<<"\t"<<fibo( i );

getch( );

return 0;

}

```

/* Function to find Fibonacci number */

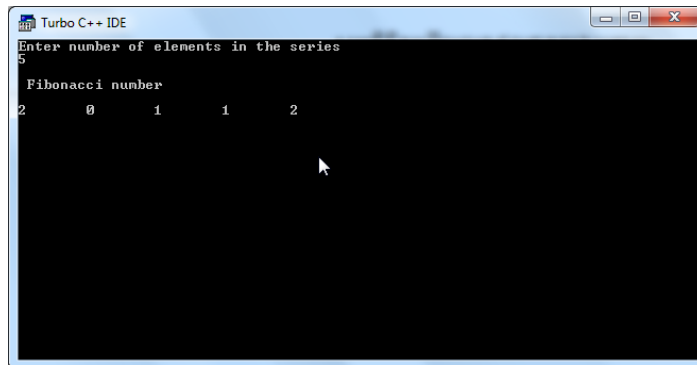
```

int fibo( int k )
{
    static int n1=1;
    static int n2=1;
    int n3;
    if( k==1 )
        n3=0;
    else if( k==2 )
        n3=1;
    else
        n3=n1+n2;
    n1=n2;
    n2=n3;
}

```

```
return(n3);  
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំណាំ **register** keyword សម្រាប់រក្សាទុកអថេរ (variables) ដែលត្រូវបានប្រកាសនៅក្នុង CPU register។

ឧទាហរណ៍១៖

សរសេរកម្មវិធីដើម្បីបង្ហាញការនៃចំនួនពីរ ១ ដល់ ១០ ដោយ ប្រើ register variable។

ចំពោះភាសា C Programming៖

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

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

```
main( )
```

```
{
```

```
clrscr( );
```

```
register int count, sqr;
```

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

```
{
```

```
    sqr = count * count;
```

```
    printf( "\n %d is %d", count, sqr );
```

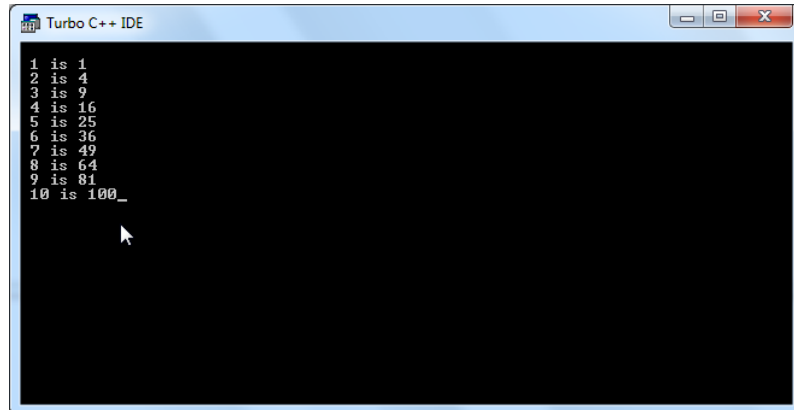
}

getch();

return 0;

}

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំពោះភាសា C++ Programming៖

```
#include<iostream.h>
```

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

```
main( )
```

```
{
```

```
clrscr( );
```

```
register int count, sqr;
```

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

```
{
```

```
    sqr = count * count;
```

```
    printf( "\n %d is %d", count, sqr );
```

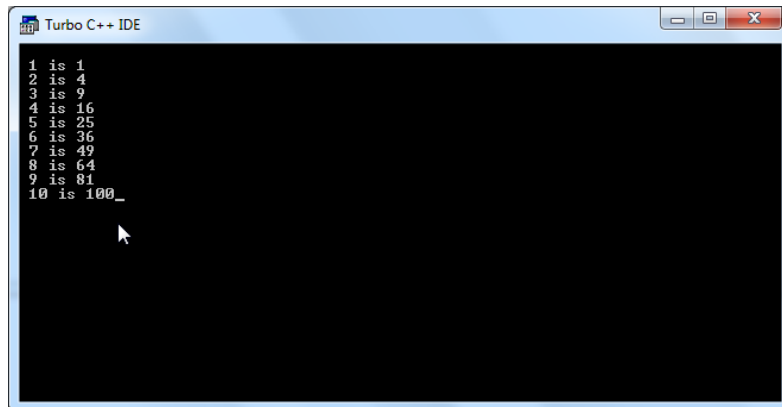
```
}
```

```
getch( );
```

```
return 0;
```

```
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ឧទាហរណ៍២៖

សូមសរសេរកម្មវិធីដើម្បីបង្ហាញពីទីតាំង address របស់អថេរ (variable) ។

ចំពោះភាសា C Programming៖

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

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

```
main( )
```

```
{
```

```
clrscr( );
```

```
register int y;
```

```
printf( "The address of y =%u\n",&y );
```

```
getch( );
```

```
return 0;
```

```
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖



ចំពោះភាសា C++ Programming៖

```
#include<iostream.h>
```

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

```
main( )
```

```
{
```

```
clrscr( );
```

```
register int y;
```

```
cout<<"The address of y =" <<y<<endl;
```

```
getch( );
```

```
return 0;
```

```
}
```

សូមពិនិត្យមើលលទ្ធផលដូចខាងក្រោម៖

