Chapter 2

Review of C++ Essentials

Lecture : SENG Sourng

Phone : (855) 92 77 12 44 / 93 77 12 44

Fb : fb.com/sourngkhmer

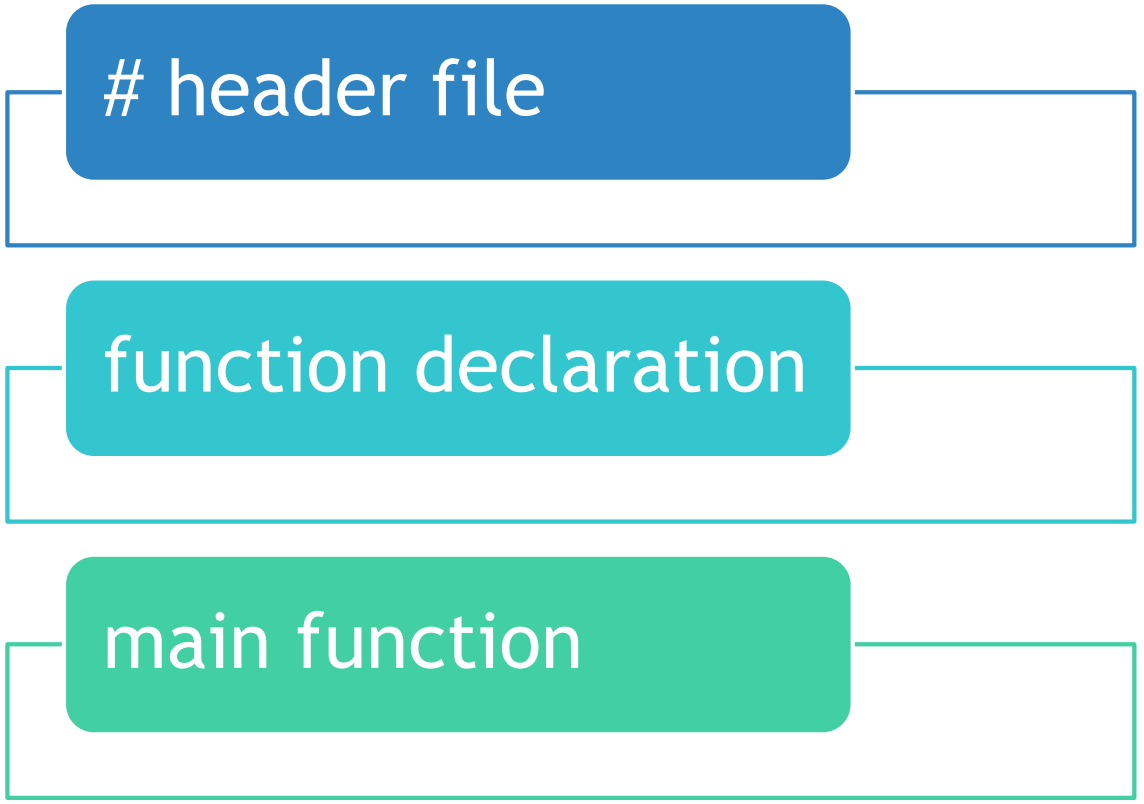
2.1. Main Program and Library Files

Preprocessor Directives

Global Data Declaration

```
int main(){
    local Data Declaration
    Statements

    return(0);
}
```

Main program function Explements



2.2. Program comments

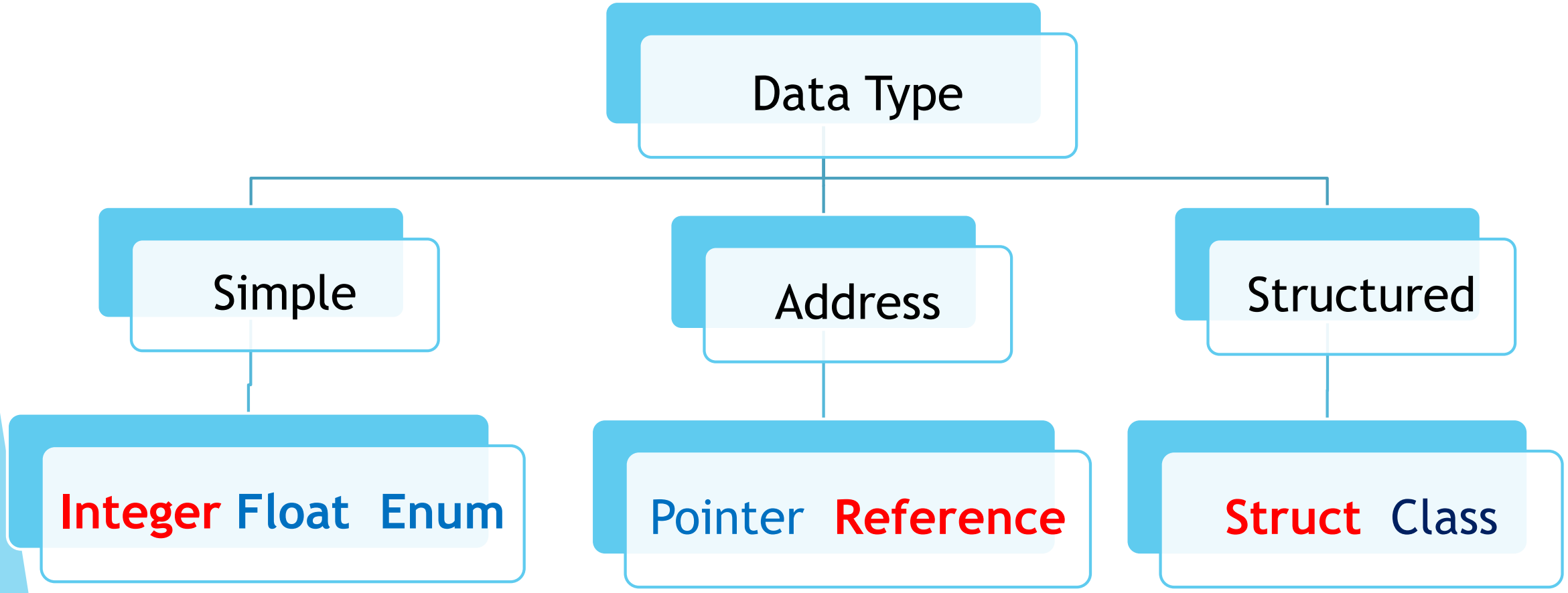
Multiple comment line

```
/*
text in comment line 1
text in comment line 2
 */
```

Single comment line

```
// text for comment line 1
// text for comment line 2
```

2.3. Data Type


Data Type
Simple
Address
Structured
Integer Float Enum
Pointer Reference
Struct Class

2.4. Simple Data Type

- int
- float
- char
- long
- double
- unsigned

2.5. Variables

Syntax

```
<data type> <list of identifiers>

<data type> <identifier> = <initial value>
```

Example :

```
int x,y,z;

float a=5.8;
```

2.5 Assignment Operator

V=E ;

- ដែល (V=variable , E=Expression)
- អថេរ E : ជាអ្នកផ្តល់តម្លៃឲ្យអថេរ V
- អថេរ V : ជា Expression, Variable , Value
- ឧទាហរណ៍ :

1. X=Y ;
2. X=20 ;
3. X=Y+10 ;
4. Y=X;

2.7.1 if statement in DATA STRUCTURE

```
if(EXP){
    // Execute Statement when EXP is true
}
```

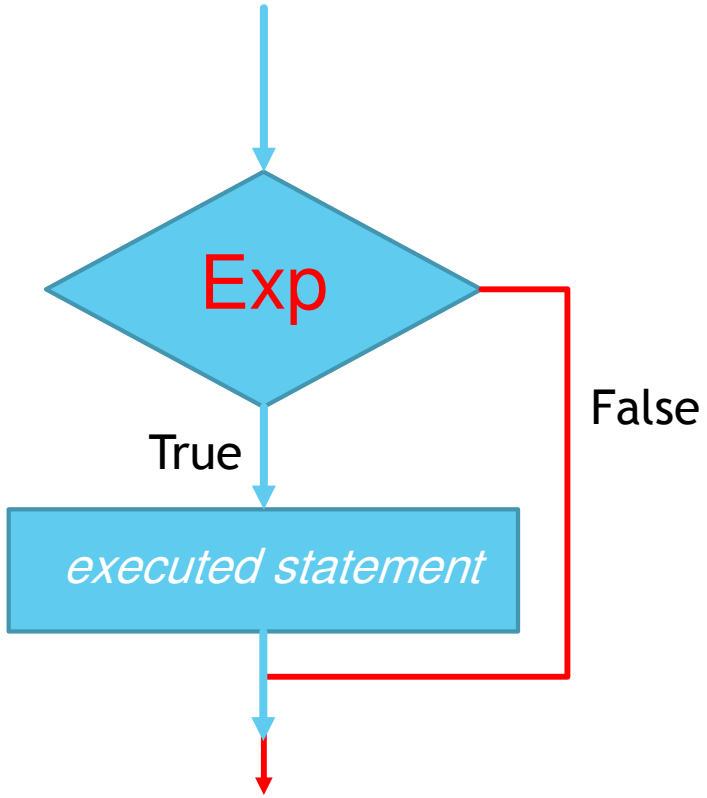
iLearned iTeach

Apply for C++ Programming

2.7.Control Statement

- 2.7.1. if statement : គឺជា statement ដែលត្រូវបានគេដាក់ឲ្យពិនិត្យលក្ខ័ណ្ឌ ថា បើពិតឲ្យវាអនុវត្តក្នុងដែននៃលក្ខ័ណ្ឌរបស់វា។ ដែននៃលក្ខ័ណ្ឌត្រូវបានគេកំណត់ដោយសញ្ញ ដង្កៀប " { .... } " ។
- រូបមន្ត Syntax :

```
if(Exp) {

    //executed statement when Exp is true

}
```



2.7.Control Statement

- 2.7.1. if statement : គឺជា statement ដែលត្រូវបានគេដាក់ឲ្យពិនិត្យលក្ខ័ណ្ឌ ថា បើពិតឲ្យវាអនុវត្តក្នុងដែននៃលក្ខ័ណ្ឌរបស់វា។ ដែននៃលក្ខ័ណ្ឌត្រូវបានគេកំណត់ដោយសញ្ញ ដង្កៀប " { .... } " ។
- រូបមន្ត Syntax :

```
if(condition) {

    //executed statement when Condition is true

}
```

```
int a=7;
if(a>5){
    cout<<“a is greater than 5”;
}
```

Example 2.7.1

2.7.2

if else statement in

DATA STRUCTURE

```
if(EXP){
        // Execute Statement when EXP is true
}else{
// Execute Statement when EXP is false
}
```

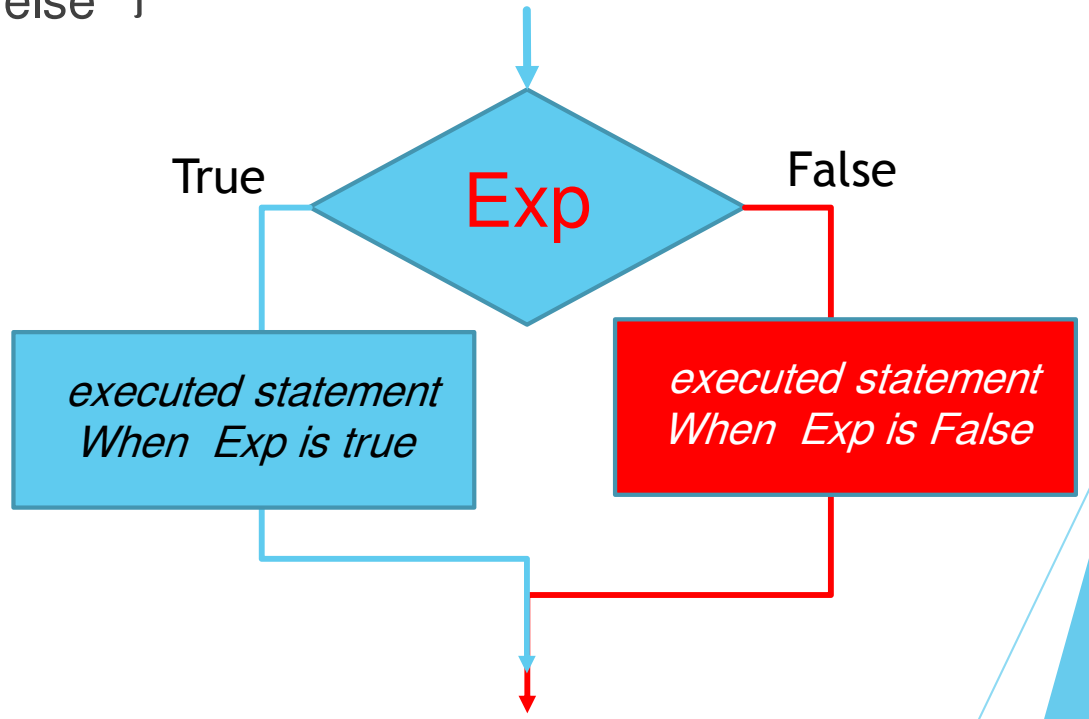
iLearned iTeach

Apply for C++ Programming

2.7.Control Statement

- 2.7.2. if...else statement : គឺជា statement ដែលត្រូវបានគេដាក់ឲ្យពិនិត្យលក្ខ័ណ្ឌ ថា បើពិតឲ្យវាអនុវត្តបណ្តា statement ដែលនៅក្នុងផ្នែកខាងក្រោម if និងជាប់នឹង if បើមិនពិតវិញ វាអនុវត្តបណ្តា statements ដែលនៅខាងក្រោម else ដែលជាប់នឹង else ។
- រូបមន្ត Syntax :

```
if(Exp) {
    //executed statement when Exp  if is true
}else{
    //executed statement when Exp  if is false
}
```



2.7.Control Statement

- 2.7.2. if...else statement : គឺជា statement ដែលត្រូវបានគេដាក់ឲ្យពិនិត្យលក្ខ័ណ្ឌ ថា បើពិតឲ្យវាអនុវត្តបណ្តា statement ដែលនៅក្នុងផ្នែកខាងក្រោម if និងជាប់នឹង if បើមិនពិតវិញ វាអនុវត្តបណ្តា statements ដែលនៅខាងក្រោម else ដែលជាប់នឹង else ។
- រូបមន្ត Syntax :

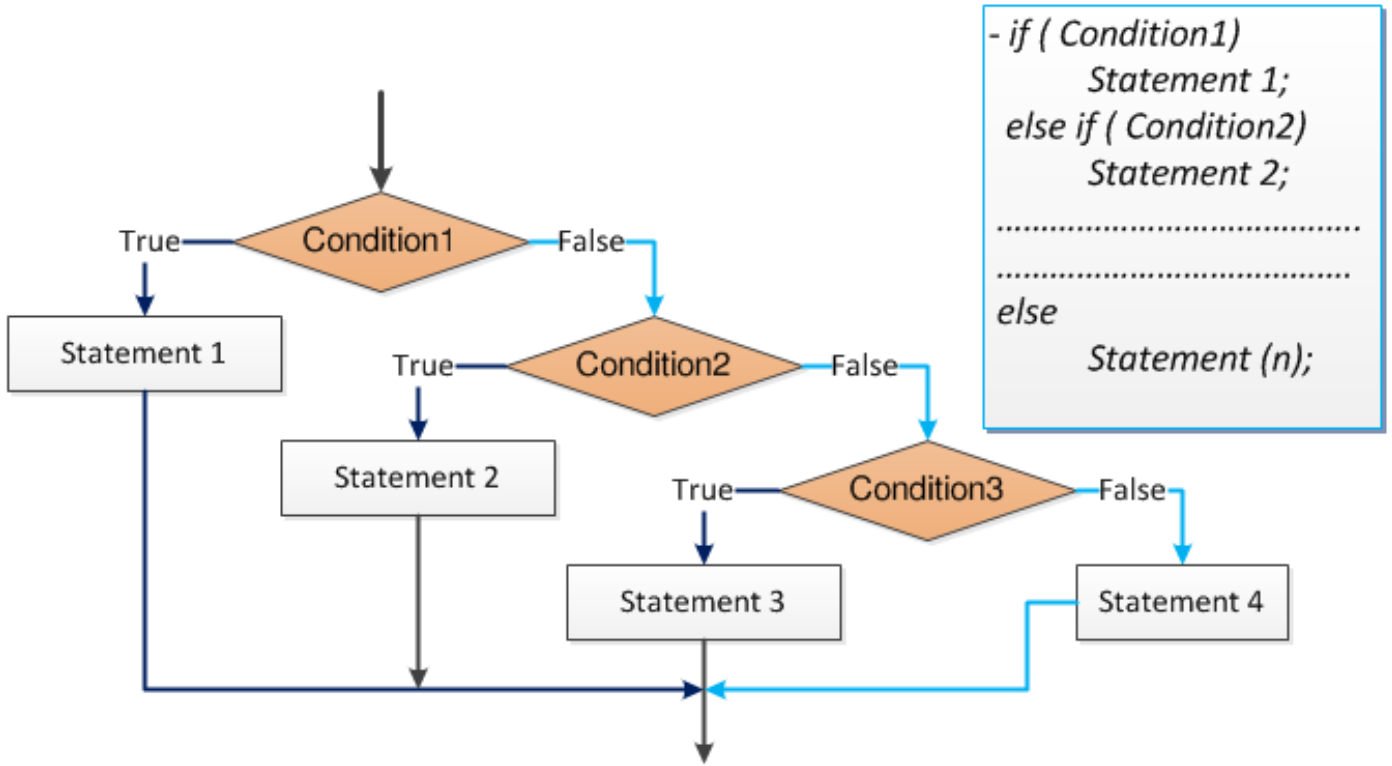
```
if(Exp) {
    //executed statement when Exp  if is true
}else{
    //executed statement when Exp  if is false
}
```

```
int a=7;
if(a>5){
    cout<<“a is greater than 5”;
}else{
    cout<<“a is less than 5”;
}
```

Example 2.7.2

2.7.Control Statement

- 2.7.3. if...elseif statement : គឺជា statements ពិសេសមួយដែលអាចធ្វើការត្រួតពិនិត្យលក្ខ័ណ្ឌច្រើនក្នុងពេលតែមួយ និងសន្សំពេលវេលាអនុវត្តទៀត។ ព្រោះថាពេលដែលលក្ខ័ណ្ឌមួយត្រូវបានអនុវត្ត ហើយឃើញថាពិត នោះលក្ខ័ណ្ឌផ្សេងទៀតត្រូវរំលង ហើយចាកចេញ។



2.7.Control Statement

- 2.7.4. Nested *if()* statement : គឺជា statements ដែលមាន Control statement if នៅក្នុង Control statement if បានន័យថា លក្ខ័ណ្ឌ if ក្នុងលក្ខ័ណ្ឌ if

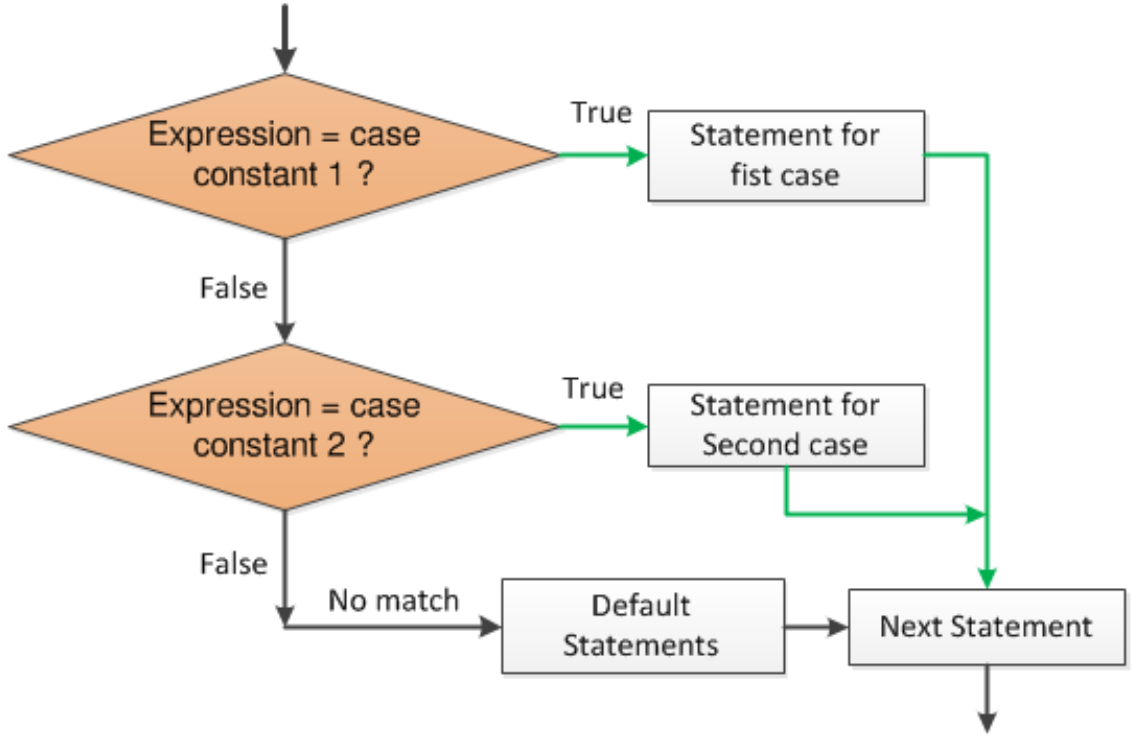
```
#include<iostream.h>
int main(){
    float score;
    cout<<"Enter Scor:";cin>>score;
    if(score>=90){
        cout<<"Grade A";
    }else if(score>=80){
        cout<<"Grade B";
    }else if(score>=70){
        cout<<"Grade C";
    }else if(score>=60){
        cout<<"Grade D";
    }else if(score>=50){
        cout<<"Grade E";
    }else{
        cout<<"Grade F, Fail !";
    }
    cout<<endl;
    return(0);
}
```

```
- if ( Condition1)
        Statement 1;
         else  ( Condition2)
        {
                if (Condition)
                        Statement 2;
                else
                {
                        Statement 3;
                        Statement 4;
                }
         }
```

2.7.Control Statement

- 2.7.4. switch statement : គឺជា statements ដែលធ្វើការអនុវត្តជម្រើសមួយក្នុងជម្រើសជាច្រើនតាមរយៈ case ។ គេតែងតែប្រើវាជាមួយ break បញ្ឈប់ការអនុវត្តពេលវាជួយ និងអនុវត្តលក្ខ័ណ្ឌណាដែលផ្ទៀងផ្ទាត់ជាមួយវា។

```
- Switch (expression)
        {
        case cont1: Statement 1 ; break;
        case cont2: Statement 2 ; break;
        .....
        .....
        default: statement n;
        }
```



Thank!

Any Questions, please?

Like, Comment and Subscribe

Chapter 3

Looping in Data Structure with C++

Lecture : SENG Sourng

Phone : (855) 92 77 12 44 / 93 77 12 44

Fb : fb.com/sourngkhmer

3.1.For loop Statement

- WHAT IS LOOP?
  - Loop are the process of statement that repeating again and again with its condition.
  - Loop is control by its condition
  - Loop there are :
    - For loop
    - While loop
    - Do ….while loop
    - For ..each loop

3.1.For loop Statement

- Syntax of for loop statement

```
for(start; expression ; increment/decrement){
    //statements to be execute when expression true
}
```

Ex_3.1:

```
for(i=1;i<5;i++){
    cout<<“Now process statement”<<i<<endl;
}
```

3.1.For loop Statement

- Example 3.1.2

```
#include<iostream.h>
int main(){
    int n,i;
    cout<<"Enter N:"; cin>>n;
    for(i=1;i<=n;i++){
        cout<<"Looping step "<<i<<endl;
        cout<<"Now i="<<i<<"is less than 5"<<endl;
    }
    cout<<"i="<<i<<" qual to "<<n<<" or greater!"<<endl;
    return(0);
}
```

3.2 while loop statement

```
#include<iostream.h>
#include<conio.h>
int main(){
    int n,i;
    cout<<"Enter N:"; cin>>n;    // n=3
    i=1; // start / intialize value
    while(i<=n){
        cout<<"Looping step "<<i<<endl;
        cout<<"Now i="<<i<<"is less than 5"<<endl;
        i++; // increment / decrement i=1+1=2+1=3+1=4
    }
    cout<<"i="<<i<<" qual to "<<n<<" or greater!"<<endl;
    getch();
    return(0);
}
```

3.2 while loop statement

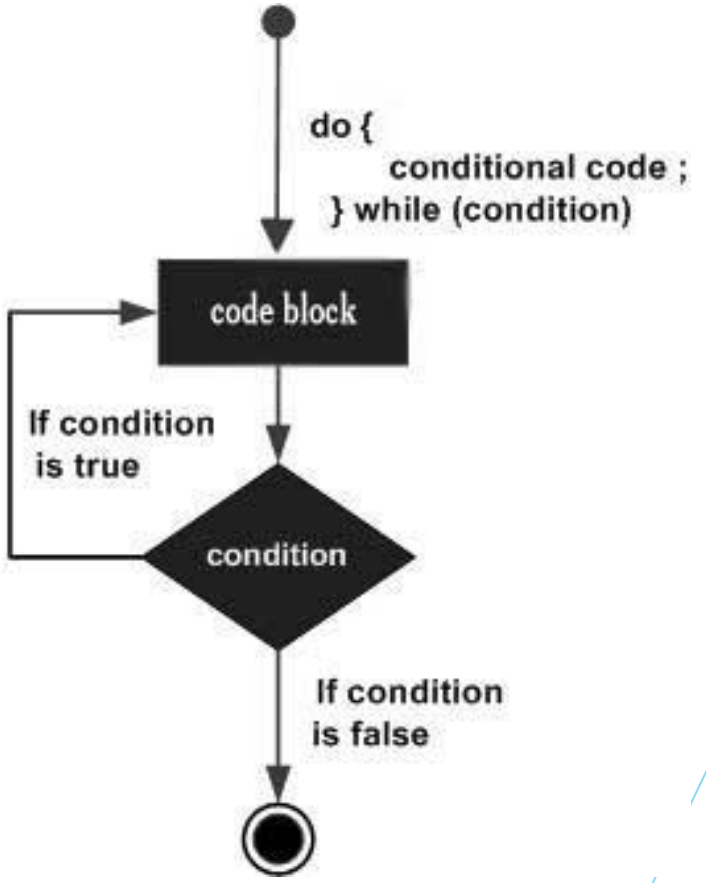
- While loop statement : គឺជា statement ពិសេសមួយដែលអាចឲ្យយើងធ្វើការអនុវត្តលើបណ្តា statements ទាំងឡាយណាដែលស្ថិតនៅក្នុង សញ្ញាដង្កៀប "{……..}" ដែលស្ថិតនៅក្រោមការគ្រប់របស់ expression។
- រូបមន្តរបស់ while គឺ:

```
Start statement
while(expression){
    //statements to be execute when expression true

    increment / decrement statement
}
```

3.3 do…while loop statement

- do…while loop statement : គឺជា statement ដែលមានលក្ខណៈស្រដៀងគ្នានឹង while loop statement ដែរ វាខុសគ្នាត្រង់ថា do…while loop វាត្រូវអនុវត្ត statements មុនពេលពិនិត្យលក្ខ័ណ្ឌ ហើយ while loop ពិនិត្យ លក្ខ័ណ្ឌមុនពេលអនុវត្ត Statements។
- រូបមន្តរបស់ while គឺ:

```
Start statement
do {
        //statements to be execute when expression true

    increment / decrement statement
}while(expression);
```



3.3 do…while loop statement

- Example 3.3.1 Using do while loop to calculate sum

```
// C++ program to add numbers until user enters 0
#include <iostream>
using namespace std;
int main() {
   float number, sum = 0.0;
    do {
      cout<<"Enter a number: ";cin>>number;
      sum += number;
      }while(number != 0.0);
      cout<<"Total sum = "<<sum;
   return 0;
}
```

Chapter 4

Function in Data Structure using C++

Lecture : SENG Sourng

Phone : (855) 92 77 12 44 / 93 77 12 44

Fb : fb.com/sourngkhmer

4.1 Introduction

- In this article, you'll learn everything about functions in C++; what type of functions are there, how to use them with examples.
- In programming, function refers to a segment that groups code to perform a specific task.
- Depending on whether a function is predefined or created by programmer; there are two types of function:
  - Library Function or Build-in functions
  - User-defined Function

4.2. Library Function

- Library functions are the built-in function in C++ programming.
- Programmer can use library function by invoking function directly; they don't need to write it themselves.
- **Example 1: Library Function**

```
#include <iostream>
#include <cmath>

using namespace std;

int main()
{
     double number, squareRoot;
     cout << "Enter a number: ";
     cin >> number;

     // sqrt() is a library function to calculate square root
     squareRoot = sqrt(number);
     cout << "Square root of " << number << " = " << squareRoot;
     return 0;
}
```

Output

```
Enter a number: 26
Square root of 26 = 5.09902
```

4.3 User Define Function

- User Define function គឺជា Function ដែលបង្កើតឡើងដោយ User (Programmer) ។ វាត្រូវបានគេបែងចែកជា៤ប្រភេទសំខាន់ៗ :
  - Function អត់មានប៉ារ៉ាម៉ែត្រ និងអត់ Return Type
  - Function អត់មានប៉ារ៉ាម៉ែត្រ និងមាន Return Type
  - Function មានប៉ារ៉ាម៉ែត្រ និង អត់មាន Return Type
  - Function មានប៉ារ៉ាម៉ែត្រ និង មាន Return Type

4.3.1 Function អត់មានប៉ារ៉ាម៉ែត្រ និងអត់ Return Type

- គឺជាFunction ដែលបង្កើតឡើងដោយ User មួយប្រភេទដែលមិនមានប្រើប្រាស់អាគុយម៉ង់(Arguments) ដែលជាប៉ារ៉ាម៉ែត្រ ហើយថែមទាំងមិនប្រើប្រាស់ ប្រភេទផ្តល់តម្លៃត្រឡប់ គឺ Return Type។
- រូបមន្ត (Syntax)

```
identifier Function Name(){
    // Processing code of function here
}
```

ត្រូវដាក់ឈ្មោះដែលទាក់ទងនឹងកូដក្នុង Function

តួដែលជាដំណោះស្រាយក្នុង Function របស់អ្នក

4.3.1 Function អត់មានប៉ារ៉ាម៉ែត្រ និងអត់ Return Type

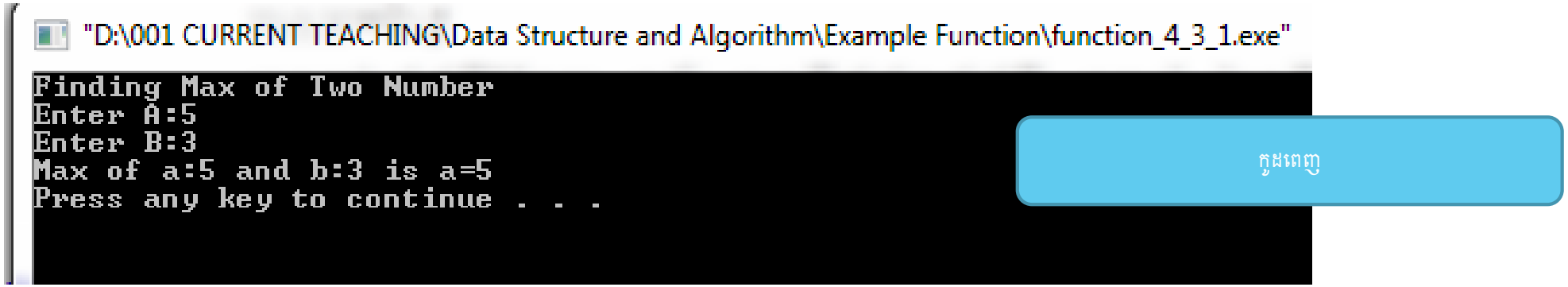
- ឧទាហរណ៍ ទី៤.៣.១ : បង្កើត function ដើម្បីគណតម្លៃធំបំផុតនៃពីចំនួន។

```
#include<iostream.h>
void findMax(){
    int a,b,max;
    cout<<"Enter A:";cin>>a;
    cout<<"Enter B:";cin>>b;
    if(a>b){
        max=a;
        cout<<"Max of a:"<<a<<" and b:"<<b<<" is a="<<max<<endl;
    }else{
        max=b;
        cout<<"Max of a:"<<a<<" and b:"<<b<<" is b="<<max<<endl;
    }
}
```

4.3.1 Function អត់មានប៉ារ៉ាម៉ែត្រ និងអត់ Return Type

```
14 int main(){
15     cout<<"Finding Max of Two Number"<<endl;
16     //call function
17     findMax();
18     return(0);
19 }
```

- លទ្ធផល : បង្កើត function ដើម្បីគណតម្លៃធំបំផុតនៃពីចំនួនគឺៗ



4.3.2 Function អត់មានប៉ារ៉ាម៉ែត្រ និងមាន Return Type

- គឺជាFunction ដែលបង្កើតឡើងដោយ User មួយប្រភេទដែលមិនមានប្រើប្រាស់អាគុយម៉ង់(Arguments) ដែលជាប៉ារ៉ាម៉ែត្រ តែមានការប្រើប្រាស់ ប្រភេទផ្តល់តម្លៃត្រឡប់ គឺ Return Type។
- រូបមន្ត (Syntax)

ត្រូវដាក់ឈ្មោះដែលទាក់ទងនឹងកូដក្នុង Function

```
Identifier Function Name(){
    // Processing code of function here
return(result);
}
```

តួដែលជាដំណោះស្រាយក្នុង Function របស់អ្នក

ប្រើប្រាស់ Keyword return ដើម្បីបោះតម្លៃពី result ដែលទទួលបាន

4.3.2 Function អត់មានប៉ារ៉ាម៉ែត្រ និងមាន Return Type

- ឧទាហរណ៍ ទី៤.៣.២ : បង្កើត function ដើម្បីគណតម្លៃធំបំផុតនៃពីចំនួន។

```
15 char findMax2(){
16          int a,b;
17          char max;
18      cout<<"Enter A:";cin>>a;
19      cout<<"Enter B:";cin>>b;
20      if(a>b){
21          max='A';
22      //  cout<<"Max betwin a:"<<a<<" and b:"<<b<<" is a="<<max<<endl;
23      }else{
24          max='B';
25      //  cout<<"Max of a:"<<a<<" and b:"<<b<<" is b="<<max<<endl;
26      }
27      return(max);
28 }
```

កូដពេញ

លំហាត់

- ចូរបង្កើត Function ដើម្បីដោះស្រាយបញ្ហានៃគណនា មេលេខគុណដោយប្រើប្រាស់ Function

- មើលគំរូដូចខាងក្រោម :
- Please Enter Number : 2

  2 x 1 = 2

  2 x 2 = 4

  2 x 3 = 6

  ----

  2 x 10 =20

4.3.3 Functionមានប៉ារ៉ាម៉ែត្រ និងអត់មាន Return Type

Function មានប៉ារ៉ាម៉ែត្រ និងអត់ Return Type មានលក្ខណៈពិសេសដោយឡែកពី Function មុនៗហើយក៏មានលក្ខណៈដូចគ្នាខ្លះដែរៗ វាមានទំរង់ទូទៅដូចខាងក្រោម៖

Identifier/Return Type Function Name(***[param1,param2,……, paramN]***){

    // Process code of function here !

}

ដែល Identifier / Return type ផ្តើមដោយពាក្យ void, int , float, char,…

និង *param1,param2,param3*, រហូតដល់ *paramN*

4.3.3 Functionមានប៉ារ៉ាម៉ែត្រ និងអត់មាន Return Type

ឧទាហរណ៍សរសេរ Function ដើម្បីគណនា ឬមើលថ្ងៃខែឆ្នាំ

```
//function definitions
void readDate(int& month, int& day, int& year)
{
  char ch; //local variable
  cout << "Enter a date (mm/dd/year): ";
  cin >> month >> ch >> day >> ch >> year;
}

void printDate(int month, int day, int year)
{
  cout << "The date is " << month << '-' << day;
  cout << '-' << year << endl;
}
```

```
//initialize automatic variables
  int mm, dd, yy;
  readDate(mm, dd, yy);
  printDate(mm, dd, yy);
```

កូដពេញ

4.3.4 Functionមានប៉ារ៉ាម៉ែត្រ និងមាន Return Type

Function មានប៉ារ៉ាម៉ែត្រ និងអត់ Return Type មានលក្ខណៈពិសេសដោយឡែកពី Function មុនៗហើយក៏មានលក្ខណៈដូចគ្នាខ្លះដែរ។ វាមានទំរង់ទូទៅដូចខាងក្រោម៖

Identifier/Return Type Function Name(*[param1,param2,……, paramN]*){

```
    // Process code of function here !

    return (value);
}
```

ដែល Identifier / Return type ផ្តើមដោយពាក្យ void, int , float, char,…

និង *param1,param2,param3*, រហូតដល់ *paramN ដែលជា Data Type ហើយនឹងអញ្ជាត*

4.3.4 Functionមានប៉ារ៉ាម៉ែត្រ និងមាន Return Type

- សរសេរកូដដើម្បីដោះស្រាយបញ្ហានៃការគណនា (ផលបូក ដក គុណ និងចែក)។តាមលក្ខ័ណ្ឌខាងក្រោម
- ពេលណាយើងបំពេញដូចខាងក្រោម :

```
int a=23,b=12; char ch=`-´;

calculateValue(a,b,ch);
```

នោះវាហ្នឹងបានតម្លៃដូចខាងក្រោម៖

Result of 23 - 12 = 11

4.3.4 Functionមានប៉ារ៉ាម៉ែត្រ និងមាន Return Type

- ដំណោះស្រាយ

```
//header files
#include<iostream.h>
//function definitions

int printResult(int a, int b, char ch)
{
    int sum;
    switch(ch){
        case '+':
            sum=a+b;
            break;
        case '-':
            sum=a-b; break;
        case '*':
            sum=a*b;break;
        case '/':
            sum=a/b;
            break;
        default:
            sum=0;
    }
    return(sum);
}
```

```
void readValue(int& a, int& b, char& ch)
{
 // char ch; //local variable
  cout << "Enter a Number (a+b): ";  cin >> a >> ch >> b;
}

int main()
{
  //initialize automatic variables
  int a, b; char ch;
  readValue(a,b,ch);
 cout<<"Result of "<<a<<ch<<b<<"="<< printResult(a, b, ch)<<endl;
  cout << "Press Enter key to exit...";
  cin.get(); cin.get(); // make DOS window stay open

  return 0;
}
```

កូដពេញ

Chapter 5

Array Structure in C++

Lecture : SENG Sourng

Phone : (855) 92 77 12 44 / 93 77 12 44

Fb : fb.com/sourngkhmer

5.1 និយមន័យ

- Array គឺជាសំនុំធាតុមួយដែលមានឈ្មោះរួមមានប្រភេទទិន្នន័យ(Data) ដូចគ្នា និងមាន Index កំណត់ទុកជាមុន។ រាល់ធាតុនិមួយៗរបស់ Array *n* ធាតុត្រូវបានផ្តុំកជាបន្តបន្ទាប់តាមលក្ខណៈរៀចទំរពីទីតាំង Index=0 រហូតដល់ទីតាំងចុងក្រោយរបស់ array គឺ *index=n-1*
- *ឧទាហរណ៍៖* ការបង្កើត array មួយដែលមាន ២០ធាតុ និងមានប្រភេទទិន្នន័យជាចំនួនគត់ int a[20] ។

តំម្លៃនៃ array

a_1	a_2	a_3	...	a_{19}	a_{20}
[0]	[1]	[2]	....	[18]	[19]

Index នៃ array

ខាងលើនេះជាប្រភេទ array មួយវិមាត្រ

5.2 ការបង្កើតarray ពីរវិមាត្រ

- Array ពីរវិមាត្រជាប្រភេទ array ដែលមានរក្សាទុកតម្លៃក្នុណៈ rows និង columns បាននិយ ថា ជាជួរឈ និងជួរដេក។
- *ឧទាហរណ៍ះ* ការបង្កើត array ពីរវិមាត្រដែលមានប្រភេទទិន្នន័យជាចំនួន ទស្សភាគ float a[3][2] ។

តម្លៃនៃ array

$A_{0,0}$	$A_{0,1}$	$A_{1,0}$	$A_{1,1}$	$A_{2,0}$	$A_{2,1}$
[0]	[1]	[2]	[3]	[4]	[5]

Index នៃ array

ខាងលើនេះជាប្រភេទ array ពីរវិមាត្រ

5.3 របៀបផ្ទុកធាតុរបស់ array

- Array គឺជា Structure អាចគណនា Address បាន(Computed address) និងមានរបៀបផ្ទុកជាបន្តបន្ទាប់តាមលក្ខណៈវិចទ័រគឺធាតុដំបូងរបស់វាគឺ a នៃ 0 ផ្ទុកត្រង់ទីតាំងដំបូង(index=0) , ធាតុ a នៃ ១ ផ្ទុកត្រង់ទីតាំងទី១ , ធាតុ a នៃ i ផ្ទុកត្រង់ទី i និង ធាតុទី a នៃ n-1 (index=n-1) ។
- ឧបមាថាយើងមាន array មួយវិមាត្រ(វិចទ័រ) a_i ដែលមាន n ធាតុហើយរាល់ធាតុនិមួយៗត្រូវការ Memory ទំហំ C word នោះ Array a_i ត្រូវការ Memory location (c x n) word ជាបន្តបន្ទាប់ និងមានរបៀបផ្ទុកដូចខាងក្រោម:

5.3 របៀបផ្ទុកធាតុរបស់ array

- ឧបមាថាយើងមាន array ពីរវិមាត្រ(វិចទ័រ) a_{ij} ដែលមាន ៣ ជួរដេក(Row) និង៤ជួរឈរ (Column) នោះយើងមានរបៀបផ្ទុករបស់វាដូចខាងក្រោម៖

	0	1	2	3
i →0	$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
a_{ij} =1	$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
2	$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$

j ↓

5.3 របៀបផ្ទុកធាតុរបស់ array

- ឧបមាថាយើងមាន array ពីរវិមាត្រ(វិចទ័រ)ដូចខាងក្រោម៖
- *int* salary[3][4];
- បញ្ចូលតម្លៃទៅឲ្យ array

  salary[1][1]=500;

  salary[2][3]=250;
- បង្ហាញតម្លៃរបស់array

  cout<<salary[2][3];

	0	1	2	3
i →0				
$salary_{ij}$ =1		500		
2				250

j ↓

5.3 របៀបផ្ទុកធាតុរបស់ array

ឧបមាថាយើងមាន array ពីរវិមាត្រ(វិចទ័រ)ដូចខាងក្រោម៖

int salary[3][4];

- បញ្ចូលតម្លៃទៅឲ្យ arrayដោយប្រើ Loop

```
for(int i=0;i<3;i++){
    for(j=0;j<4;j++{
        cout<<“salary[“<<i<<“][“<<j<<“]:”;
        cin>>salary[i][j];
    }
}
```

i →0

$salary_{ij}$ =1

2

0	1	2	3
	500		
			250

j ↓

5.3 របៀបផ្ទុកធាតុរបស់ array

- បង្ហាញតម្លៃនៃ array *int* salary[3][4] ដោយប្រើ Loop

```
for(int i=0;i<3;i++){
    for(j=0;j<4;j++{
        cout<<“salary[“<<i<<“][“<<j<<“]:”;
        cout<<alary[i][j];
    }
cout<<endl;
}
```

5.4.របៀបគណនា Address របស់ Array

(១). Array មួយវិមាត្រ (one dimensional array)

- Address នៃធាតុ (a_i)ត្រូវបានគណនាតាមរូបមន្ត៖

$$Loc(a_i) = Lo + C*i$$

- $Loc(a_i)$ ជា address នៃធាតុ a_i
- Lo ជា address ដើម(address នៃធាតុ a_0)
- i ជា index
- C ជាទំហំនៃធាតុដើមនីមួយៗរបស់ array ដែលចាប់យក(គិតជា word)

5.4.របៀបគណនា Address របស់ Array

- ឧទាហរណ៍១៖

  float a[20]; 4byes=2words

  int a[50]; 2 byes=1words

- ឧទាហរណ៍២៖

- គេមាន array float a[45]; បើ address ដើមមានតម្លៃ១៨២ ចូរគណនា address នៃធាតុទី១៨។

ដំណោះស្រាយឧទាហរណ៍ទី២

- គណនា address នៃធាតុ a_{18}
- តាមរូបមន្ត $Loc(a_i)=Lo+C * i$
- នាំឲ្យ $Loc(a_{18})=182+2 *18= 218$
- *float=4bytes , int=2bytes ហើយ*
- *1 word=2bytes → float=2words*

លំហាត់

- ឧទាហរណ៍៣៖
  - គេមាន array float a[24]; បើaddress នៃធាតុទី១៨=២១៥ តើ address ដើមមានតម្លៃ?។

5.4.របៀបគណនា Address របស់ Array

(២). Array ពីរវិមាត្រ (two dimensional array)

- Address នៃធាតុ (a_{ij})ត្រូវបានគណនាតាមរូបមន្ត៖

$$Loc(a_{ij}) = Lo + C*(n*i+j)$$ Row major order

$$Loc(a_{ij}) = Lo + C*(m*j+i)$$ Column major order

- ដែល m = row និង n = column

ឧទាហរណ៍

- គេមាន array : int a[20][15]; បើ address ដើមមានតម្លៃ 246 ចូរគណនា address នៃធាតុ a_{79} តាមរយៈ Row Major Order និង Column major order ។

មេរៀនទី៦

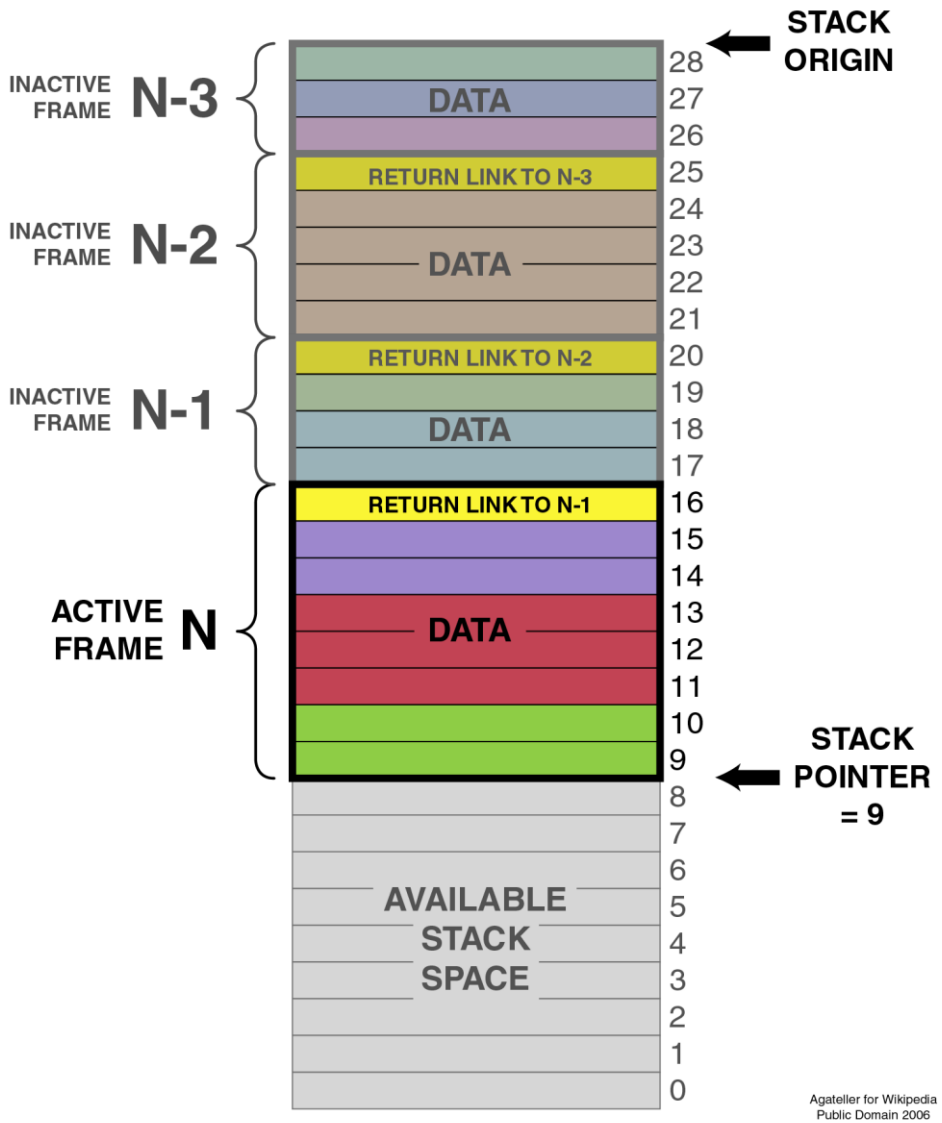
សិក្ខាអំពី STACK

ក្នុងមេរៀននេះអ្នកនឹងសិក្សាអំពីរបៀបដំណើរការនៃ Stack និង របៀបផ្ទុកធាតុរបស់ Stack

៦.១ Stack

- **និយមន័យ៖** Stack គឺជា list ពិសេសមួយដែលរាល់ប្រមាណវិធីបន្ថែមធាតុ(push) និងរាល់ប្រមាណវិធីដកយកធាតុ(pop)របស់ stack ត្រូវបានអនុវត្តនៅផ្នែកម្ខាងនៃកំពូល stackត្រូបានហៅថា Top។ ម្យ៉ាងទៀត Stack ត្រូវបានហៅថាទម្រង់ list ដែលមានលក្ខណៈ LIFO (Last In First Out)មានន័យថាធាតុចូលក្រោយត្រូវបានយកចេញមុន។

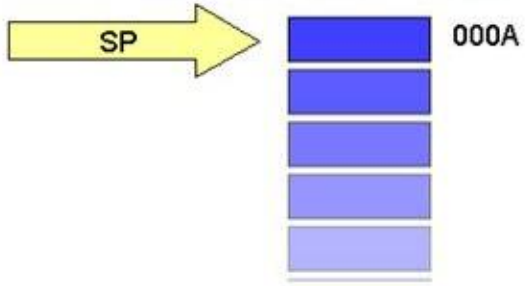
๖.๑ Stack



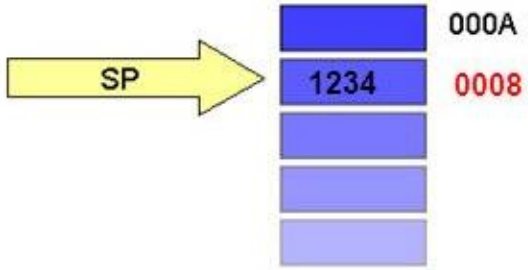
Agateller for Wikipedia
Public Domain 2006

๖.๑ Stack

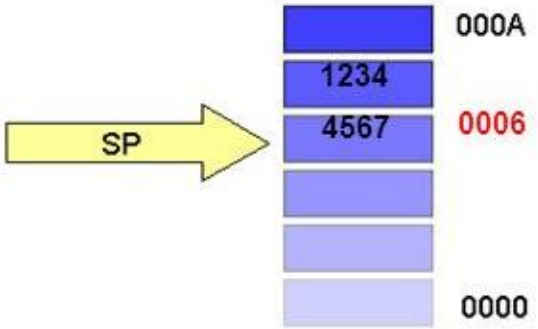
Empty Stack. Stack pointer points to highest address allocated for stack. As elements are added, it decrements. SP = 000A



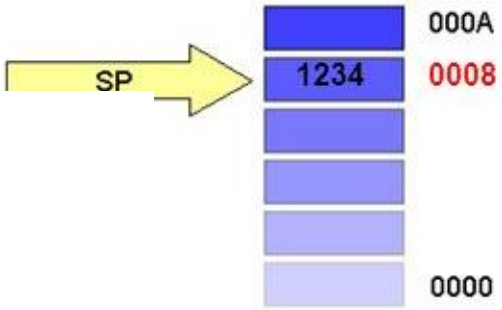
One 16 bit element pushed on stack. Stack pointer = 0008



Two 16 bit elements pushed on stack. Stack pointer = 0006



On element popped. Stack pointer = 0008



៦.១ Stack

- យើងមាន Structure ទូទៅរបស់ Stack ដូចខាងក្រោម:

LIFO : (Last In First Out)

- ## ការផ្ទុក Stack ជា Array
- គេអាចផ្ទុក stack លើ Array មួយវិមាត្រដែលមាន n ផ្នែកនៃ Memory បន្តបន្ទាប់គ្នា បើ address នៃផ្នែក Top របស់ stack តាងដោយ Top នោះគេបានដូចរូបរាងខាងក្រោម:

S_1	S_2	S_3	S_4	S_5	...	S_{n-2}	S_{n-1}
					...	S	

៦.១. Stack

- ការបង្កើត Data Structure សម្រាប់ Stack

```
#define max stack …

typedef int Element Type;

struct stack type

{

        int Top;
        element type s [maxstack];

};
```

- បើ st ជា Stack នោះត្រូវ declare: Stack Type st;

៦.២ ការប្រើប្រាស់លើ Stack

- Stack ត្រូវបានប្រើសម្រាប់រាប់ដោះស្រាយបញ្ហា ដែលមានលក្ខណៈ LIFO(last in first out)
  - Stack ត្រូវបានគេប្រើប្រាស់សម្រាប់ការងារ compiler របស់ Machine ។
  - Stack ត្រូវបានគេប្រើសម្រាប់ដោះស្រាយបញ្ហាតាមបែប Recursive មានន័យថាវាអាចជំនួសអោយការងារ Recursive ។
  - Stack ត្រូវបានគេប្រើសម្រាប់អនុវត្តការងារ call function នៅក្នុង Program។

៦.៣.ប្រមាណវិធីលើ Stack

- យើងមានប្រមាណវិធីមួយចំនួនរបស់ Stack ដូចខាងក្រោម
- របៀបបង្កើត Data Structure
  - Stack ត្រូវបានប្រកាសជាទំរង់ struct ដែលរួមមាន 2 field:
    - ✓ Field Top ជាអថេរសម្រាប់ចង្អុលទៅកាន់ធាតុស្ថិតនៅកំពូល Stack
    - ✓ Field Node សម្រាប់ផ្ទុក Data នីមួយៗរបស់ stack ទម្រង់ list របស់ stack ត្រូវបានអនុវត្តជាមួយ Array & Pointer ប៉ុន្តែមេរៀននេះយើងប្រើ Array:

៦.៣.ប្រមាណវិធីលើ Stack

```
#define Max stack

struc stack

{

   int Top;

   Data type Node [Maxstack];

};

struct stack st, a [100],*pt;
```

៦.៣.ប្រមាណវិធីលើ Stack

- កម្មវិធី Initialize: សម្រាប់បង្កើត stack ទទេរគឺ: Top = -1
- ប្រមានវិធី Full stack : សម្រាប់ត្រួតពីនិត្យមើល stack ពេញគឺ Top = Max stack – 1
- ប្រមាណវិធី Push : សម្រាប់បន្ថែមធាតុ Item ណាមួយចូល stack ត្រង់ទីតាំង Top ដោយអនុវត្តតាមពីរដំណាក់កាលគឺ:
  - ប្រសិន full stack នោះបង្ហាញប្រាប់ stack overflow
  - ប្រសិន not full stack នោះ
    - បង្កើត Top មួយទីតាំងគឺ Top = Top + 1
    - ផ្ទេតម្លៃ Item ចូល stack ត្រង់ទីតាំង
      - Top : Node [Top] = Item;

៦.៣.ប្រមាណវិធីលើ Stack : ឧទាហរណ៍

```
// stack::push/pop
#include <iostream>       // std::cout
#include <stack>          // std::stack
using namespace std;
int main ()
{
  stack<int> mystack;
  for (int i=0; i<5; ++i)
    mystack.push(i);

  cout << "Popping out elements...";
  while (!mystack.empty())
  {
     cout << ' ' << mystack.top();
     mystack.pop();
  }
  cout << '\n';
  return 0;
}
```

- នេះជាឧទាហរណ៍នៃការបង្កើត Stack ដែលមានការបញ្ចូល នឹងដកយកធាតុចេញវិញ។
- នៅក្នុង C++ គឺការបង្កើត stack ត្រូវប្រើប្រាស់ keyword stack ដែលមានដូចក្នុងឧទាហរណ៍។
- mystack.push(i) : គឺសំរាប់បញ្ចូលធាតុទៅឲ្យ stack
- mystack.top() : គឺសំរាប់រំកិលចំនុចរបស់ stack pointer ទៅកាន់ ផ្នែកលើគេ។
- mystack.pop() : គឺប្រើដើម្បីលុប