មេរៀនទី៥៖

Array និង String

១. លក្ខណៈទូទៅរបស់Array៖

១.១ តើ Array គឺជាអ្វី?

Array គឺជាលំដាប់តួនៃការរៀបចំតំបន់របស់ RAM បន្តបន្ទាប់គ្នា ដើម្បីផ្ទុកនូវ តំលៃជាច្រើនដែលមានប្រភេទទិន្នន័យដូចគ្នា។ ការតាង Array នៃប្រភេទទិន្នន័យមួយគឺអាចជំនួសអោយអថេរជាច្រើនដែលមានប្រភេទទិន្នន័យដូចគ្នា។ ម៉្យាងវិញទៀត Array ស្ថិតនៅក្នុងប្រភេទទិន្នន័យពិសេសដែលប្រើប្រាស់តំបន់របស់ RAM តាមលំដាប់។

១.២ ការប្រើប្រាស់ Array មួយវិមាត្រ៖

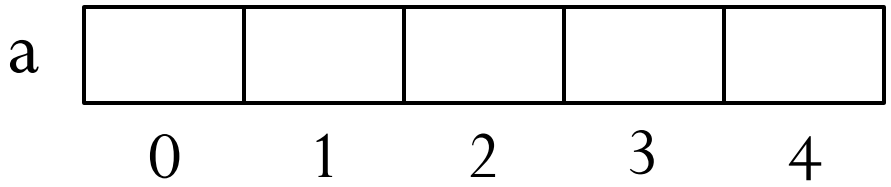
Array មួយវិមាត្រគឺជា Array ដែលមាន Index តែមួយសំរាប់តំណាងអោយសន្ទស្សន៍នៃជួរដេក ដើម្បីកំណត់ធាតុនីមួយៗរបស់ Array នោះ។ ដើម្បីប្រកាស Array មួយវិមាត្រ យើងត្រូវអនុវត្តតាមទំរង់ដូចខាងក្រោម៖

Datatype ArrayName[NumberOfElements];

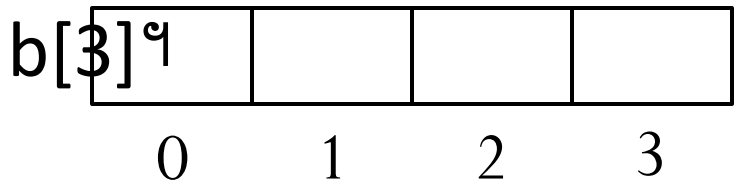
ដែល NumberOfElements គឺជាចំនួនធាតុរបស់ Array មួយវិមាត្រដោយយើងត្រូវកំណត់ជាចំនួនគត់វិជ្ជមាន ហើយមានតំលៃចាប់ពី 1 ឡើងទៅ។

ឧទាហរណ៍៖

int a[5]; មាននិយមថា a គឺជា Array នៃចំនួនគត់ Integer ដែលមាន៥ធាតុគឺ a[0], a[1], a[2], a[3], a[4]។



float b[4]; មាននិយមថា b គឺជា Array នៃចំនួនទសភាគ Float ដែលមាន៤ធាតុគឺ b[0], b[1], b[2], b[3]។



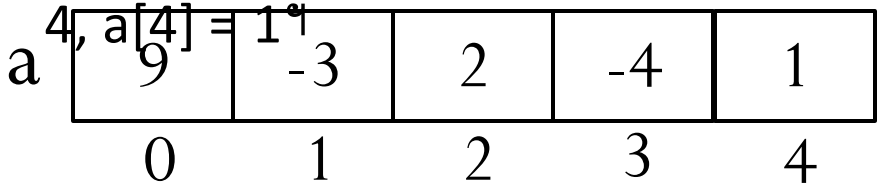
ចំណាំ៖ យើងអាចកំណត់តំលៃអោយធាតុនីមួយៗរបស់ Array នៅពេលប្រកាសក្នុងទ្រង់ទ្រាយដូចខាងក្រោម៖

Datatype ArrayName[NumberOfElements] = {Value1, …, ValueN};

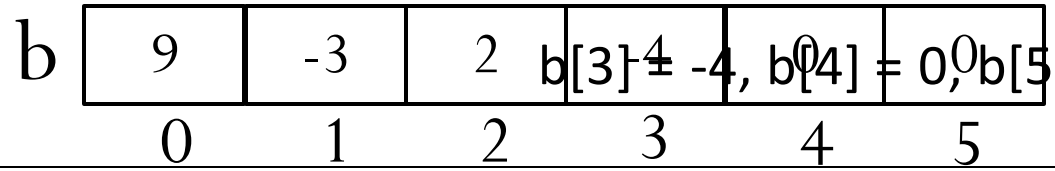
ដែល NumberOfElements ត្រូវកំណត់តំលៃអោយធំជាងឬស្មើ N។ បើ NumberOfElements ធំជាង N នាំអោយធាតុដែលនៅសល់មានតំលៃស្មើសូន្យ។

ឧទាហរណ៍៖

int a[5] = {9, -3, 2, -4, 1}; មានន័យថា a[0] = 9, a[1] = -3, a[2] = 2, a[3] = -4, a[4] = 1។



int b[6] = {9, -3, 2, -4}; មានន័យថា b[0] = 9, b[1] = -3, b[2] = 2, b[3] = -4, b[4] = 0, b[5] = 0។



១.២.១. របៀបបញ្ចូលតំលៃអោយធាតុនីមួយៗរបស់ Array

មួយវិមាត្រ៖

យើងអាចបញ្ចូលតំលៃអោយធាតុនីមួយៗរបស់ Array មួយវិមាត្រតាមទ្រង់ទ្រាយដូចខាងក្រោម៖

```
int  i, n, a[100];
```

បញ្ចូលចំនួនធាតុរបស់ Array (n) ចាប់ពី១ ដល់ ១០០

```
for(i = 0; i < n; i++){
        printf(“Input a[%d] = ”, i);
        scanf(“%d”, &a[i]);
}
```

១.២.២. របៀបបញ្ចេញតំលៃនៃធាតុនីមួយៗរបស់ Array មួយវិមាត្រ៖

យើងអាចបញ្ចេញតំលៃនៃធាតុនីមួយៗរបស់ Array មួយវិមាត្រតាមទ្រង់ទ្រាយដូចខាងក្រោម៖

```
for(i = 0; i < n; i++)
        printf("a[%d] = %d\n", i, a[i]);
```

ឬ

```
for(i = 0; i < n; i++)
        printf("%d\t", a[i]);
```

ឧទាហរណ៍៖ ចូរសរសេរប្រូក្រាមមួយសំរាប់បញ្ចូលតំលៃអោយ Array មួយវិមាត្រ ដោយយើងត្រូវបញ្ចូលចំនួនធាតុត្រូវមានតំលៃធំជាង២ និងតូចជាងឬស្មើ ១០០។ ជាចុងក្រោយបង្ហាញតំលៃទាំងនោះក្នុងមួយជួរឈរឬក្នុងមួយជួរដេក។

```
#include<stdio.h>

#include<conio.h>

#define   P  printf

#define   S  scanf

void  main(){   int  i, n;

                int a[100];

Back:     clrscr();

          P("\n Input Number of Elements = "); S("%d", &n);

          if(n<2 || n>100) goto  Back;
```

```
        for(i = 0; i < n; i++){
                P("Input a[%d] = ", i);
                S("%d", &a[i]);
        }
        P("All Elements of Array are:\n");
        for(i = 0; i < n; i++)
                P("a[%d] = %d\n", i, a[i]);
        getch();
}
```

- ដើម្បីគណនាផលបូកសរុបនៃតំលៃរបស់ Array មួយវិមាត្រួច គណនាមធ្យមភាគ យើងត្រូវអនុវត្តតាមទ្រង់ទ្រាយដូចខាងក្រោម៖

```
long  t = 0;
float   avg;
for(i = 0; i < n; i++)
        t = t + a[i];
avg = float(t) / n;
P(“Total = %ld\n”, t);
P(“Average = %0.2f\n”, avg);
```

- ដើម្បីរកតំលៃតូចបំផុត (Minimum) និងតំលៃធំបំផុត (Maximum) នៃតំលៃរបស់ Array មួយវិមាត្រ យើងត្រូវអនុវត្តតាមទ្រង់ទ្រាយដូចខាងក្រោម៖

```
int  min, max;
min = a[0];     max = a[0];
for(i = 1; i < n; i++){
        if(min>a[i]) min = a[i];
        if(max<a[i]) max = a[i];
}
P("Minimum = %d\nMaximum = %d\n", min, max);
```

- ដើម្បីតំរៀបតំលៃរបស់ Array មួយវិមាត្រតាមលំដាប់កើន (Ascending) យើងត្រូវអនុវត្តតាមទ្រង់ទ្រាយដូចខាងក្រោម៖

```
int  j, temp;
for(i = 0; i < n-1; i++)
        for(j = i+1; j < n; j++)
                if(a[i]>a[j]){
                        temp = a[i];    a[i] = a[j];  a[j] = temp;
                }
P("All Elements of Array after sorting by Ascending:\n");
for(i = 0; i < n; i++)
        P("%d\t", a[i]);
```

១.២.៣. ការប្រើប្រាស់ Array មួយវិមាត្រធ្វើជា Parameter របស់អនុគមន៍៖

យើងអាចប្រើ Array មួយវិមាត្រធ្វើជា Parameter របស់អនុគមន៍មួយចំនួនដូចខាងក្រោម៖

ក. អនុគមន៍សំរាប់បញ្ចូលតំលៃអោយធាតុនីមួយៗរបស់ Array មួយវិមាត្រមានទ្រង់ទ្រាយដូចខាងក្រោម៖

```
void  inputArray(char  arr, int  x[], int  m){
        for(int i = 0; i < m; i++){
                P(“Input %c[%d] = ”, arr, i);
                S(“%d”, &x[i]);
        }
}
```

យើងអាចហៅអនុគមន៍ខាងលើ ដើម្បីបញ្ចូលតំលៃអោយធាតុនីមួយៗរបស់ Array មួយវិមាត្រដោយប្រើ Call Statement ដូចខាងក្រោម៖

inputArray(‘a’, a, n);

ដែល៖

- a គឺជាឈ្មោះរបស់ Array មួយវិមាត្រដែលយើងបានប្រកាសរួចគឺ int a[100];
- n គឺជាចំនួនធាតុដែលយើងបានប្រកាសនិងបញ្ចូលតំលៃចាប់ពី២ រហូតដល់ ១០០។

ខ. អនុគមន៍សំរាប់បញ្ចេញតំលៃនៃធាតុនីមួយៗរបស់ Array មួយវិមាត្រមានទ្រង់ទ្រាយដូចខាងក្រោម៖

```
void  outputArray(int  x[], int  m){
        for(int i = 0; i < m; i++)
                P(“%d\t”, x[i]);
}
```

យើងអាចហៅអនុគមន៍ខាងលើ ដើម្បីបញ្ចេញតំលៃនៃធាតុនីមួយៗរបស់ Array មួយវិមាត្រដោយប្រើ Call Statement ដូចខាងក្រោម៖

```
outputArray(a, n);
```

គ. អនុគមន៍សំរាប់រកតំលៃតូចបំផុតនិងធំបំផុតនៃធាតុរបស់ Array មួយវិមាត្រមានទ្រង់ទ្រាយដូចខាងក្រោម៖

```
void  findMinMaxArray(int  x[], int  m){
        int  min, max;
        min = x[0];  max = x[0];
        for(int i = 1; i < m; i++){
                if(min> x[i]) min = x[i];
                if(max<x[i]) max = x[i];
        }
        P("Minimum is %d\nMaximum is %d\n", min, max);
}
```

យើងអាចហៅអនុគមន៍ខាងលើ ដើម្បីបញ្ចេញតំលៃនៃធាតុនីមួយៗរបស់ Array មួយវិមាត្រដោយប្រើ Call Statement ដូចខាងក្រោម៖

```
findMinMaxArray(a, n);
```

ឃ. អនុគមន៍សំរាប់ស្វែងរកតំលៃណាមួយនៅក្នុងគ្រប់ធាតុរបស់ Array មួយវិមាត្រមានទ្រង់ទ្រាយដូចខាងក្រោម៖

```
void  searchArray(int  x[], int  m, int p){
        int  search = 0;
        for(int i = 0; i < m; i++){
                if(p ==x[i]){
                        P("%d at position %d\n", p, i);
                        search = 1;
                }
        }
        if(search==0)
                P("%d not found in the list of Array.\n", p);
}
```

យើងអាចហៅអនុគមន៍ខាងលើ ដើម្បីបញ្ចេញតំលៃនៃធាតុនីមួយៗរបស់ Array មួយវិមាត្រដោយប្រើ Call Statement ដូចខាងក្រោម៖

```
searchArray(a, n, q);
```

```
#include<stdio.h>
#include<conio.h>
#define   P  printf
#define   S  scanf
void  inputArray(char  arr, int  x[], int  m){
        for(int i = 0; i < m; i++){
                P(“Input %c[%d] = ”, arr, i);
                S(“%d”, &x[i]);
        }
}

void  outputArray(int  x[], int  m){
        for(int i = 0; i < m; i++)
                P(“%d\t”, x[i]);
}
```

```
void  findMinMaxArray(int  x[], int  m){
        int  min, max;
        min = x[0];  max = x[0];
        for(int i = 1; i < m; i++){
                if(min> x[i])  min = x[i];
                if(max<x[i])  max = x[i];
        }
        P("Minimum is %d\nMaximum is %d\n", min, max);
}
void  searchArray(int  x[], int  m, int p){
        int  search = 0;
        for(int i = 0; i < m; i++){
                if(p ==x[i]){
                        P("%d at position %d\n", p, i);
                        search = 1;
                }
        }
        if(search==0)
                P("%d not found in the list of Array.\n", p);
}
```

```
void  main(){
        int  i, j, n, temp, a[100], b[100];
Back:   clrscr();
        P("Input Number of Elements = "); S("%d", &n);
        if(n<2 || n>100) goto  Back;
        P("Input All Elements of Array A:\n");
        inputArray('a', a, n);
        P("Input All Elements of Array B:\n");
        inputArray('b', b, n);
        P("All Elements of Array A are:\n");
        outputArray(a, n);
        P("\nAll Elements of Array B are:\n");
        outputArray(b, n);
```

```
for(i = 0; i < n-1; i++)
        for(j = i+1; j < n; j++)
                if(a[i]>a[j]){
                        temp = a[i];     a[i] = a[j];  a[j] = temp;
                }
P("\nAll Elements of Array A after sorting by Ascending:\n");
outputArray(a, n);
for(i = 0; i < n-1; i++)
        for(j = i+1; j < n; j++)
                if(b[i]<b[j]){
                        temp = b[i];     b[i] = b[j];  b[j] = temp;
                }
P("\nAll Elements of Array B after sorting by Descending:\n");
outputArray(b, n);
```

```
        P("\nMinimum and Maximum in Array A:\n");
        findMinMaxArray(a, n);
        P("Minimum and Maximum in Array B:\n");
        findMinMaxArray(b, n);
        int  q;
        P("Input Number for Searching:");
        S("%d", &q);
        P("Search number in Array A:\n");
        searchArray(a, n, q);
        P("Search number in Array B:\n");
        searchArray(b, n, q);
        getch();
}
```

១.៣. ការប្រើប្រាស់ Array ពីរវិមាត្រ

Array ពីរវិមាត្រគឺជា Array ដែលមាន Index ចំនួនពីរសំរាប់តំណាងអោយសន្ទស្សន៍នៃជួរដេកនិងជួរឈរ ដើម្បីកំណត់ធាតុនីមួយៗរបស់ Array នោះ។ ដើម្បីប្រកាស Array ពីរវិមាត្រ យើងត្រូវអនុវត្តតាមទំរង់ដូចខាងក្រោម៖

Datatype ArrayName[Rows][Columns];

ដែល Rows គឺជាចំនួនជួរដេក ដោយយើងត្រូវកំណត់ជាចំនួនគត់វិជ្ជមាន ហើយមានតំលៃចាប់ពី 1 ឡើងទៅ។ ចំណែកឯ Columns គឺជាចំនួនជួរឈរ ដោយយើងត្រូវកំណត់ជាចំនួនគត់វិជ្ជមាន ហើយមានតំលៃចាប់ពី 1 ឡើងទៅ។

ឧទាហរណ៍៖

int a[4][3]; មានន័យថា a គឺជា Array ពីរវិមាត្រសំរាប់ផ្ទុកតំលៃនៃចំនួនគត់ Integer ដែលមាន១២ធាតុដូចជា a[0][0], a[0][1], a[0][2], a[1][0], a[1][1], a[1][2], a[2][0], a[2][1], a[2][2], a[3][0], a[3][1], a[3][2]។

	0	1	2
0	a[0][0]	a[0][1]	a[0][2]
1	a[1][0]	a[1][1]	a[1][2]
2	a[2][0]	a[2][1]	a[2][2]
3	a[3][0]	a[3][1]	a[3][2]

១.៣.១. របៀបបញ្ចូលតំលៃអោយធាតុនីមួយៗរបស់ Array ពីរវិមាត្រ:

យើងអាចបញ្ចូលតំលៃអោយធាតុនីមួយៗរបស់ Array ពីរវិមាត្រតាមទ្រង់ទ្រាយដូចខាងក្រោម:

```
int  i, j, row, col, a[10][10];
```

បញ្ចូលចំនួនជួរដេក (row) ចាប់ពី១ ដល់ ១០

បញ្ចូលចំនួនជួរឈរ (col) ចាប់ពី១ ដល់ ១០

```
for(i = 0; i < row; i++){
    for(j = 0; j <col; j++){
            P("Input a[%d][%d] = ", i, j);
            S("%d", &a[i][j]);
    }
}
```

១.៣.២. របៀបបញ្ចេញតំលៃនៃធាតុនីមួយៗរបស់ Array ពីរវិមាត្រ៖

យើងអាចបញ្ចេញតំលៃនៃធាតុនីមួយៗរបស់ Array ពីរវិមាត្រ តាមទ្រង់ទ្រាយដូចខាងក្រោម៖

+ បញ្ចេញក្នុងទ្រង់ទ្រាយធម្មតា៖

```
for(i = 0; i < row; i++){
    for(j = 0; j <col; j++){
        P("a[%d][%d] = %d\n", i, j, a[i][j]);
    }
}
```

+ បញ្ចេញក្នុងទ្រង់ទ្រាយជាម៉ាទ្រីស (Matrix)៖

```
P(“Matrix  A:\n”);
for(i = 0; i < row; i++){
        for(j = 0; j <col; j++){
                P(“%d\t”, a[i][j]);
        }
        P(“\n”);
}
```

ឧទាហរណ៍៖ ចូរសរសេរប្រូក្រាមមួយសំរាប់បញ្ចូលតំលៃអោយ Array ពីរវិមាត្រ ដោយយើងត្រូវបញ្ចូលចំនួនជួរដេកនិងចំនួនជួរឈរចាប់ពី១ដល់១០។ ជាចុងក្រោយបង្ហាញតំលៃទាំងនោះក្នុងទំរង់ជាម៉ាទ្រីស។

```
#include<stdio.h>
#include<conio.h>
#define   P  printf
#define   S  scanf
void  main(){
          int  i, j, row, col, a[10][10];
Back1:    clrscr();
          P(“Input Number of Rows = ”); S(“%d”, &row);
          if(row<1 || row>10) goto  Back1;
```

```
Back2:
        P("Input Number of Columns = "); S("%d", &col);
        if(col<1 || col>10) goto  Back2;
        for(i = 0; i < row; i++){
                for(j = 0; j < col; j++){
                        P("Input  a[%d][%d] = ", i, j);
                        S("%d", &a[i][j]);
                }
        }
        P("Matrix  A:\n");
        for(i = 0; i < row; i++){
                for(j = 0; j < col; j++){
                        P("%d\t", a[i][j]);
                }
                P("\n");
        }
        getch();
}
```

១.៣.៣. របៀបគណនាផលបូករវាងម៉ាទ្រីស៖

យើងអាចគណនាផលបូករវាងម៉ាទ្រីស a[10][10] ជាមួយនឹងម៉ាទ្រីស b[10][10] តាមទ្រង់ទ្រាយដូចខាងក្រោម៖

- បញ្ចូលតំលៃ row និង col ចាប់ពី១ដល់១០។
- បញ្ចូលតំលៃអោយម៉ាទ្រីស a ដែលមានចំនួនជួរដេក row ហើយចំនួនជួរឈរ col។
- បញ្ចូលតំលៃអោយម៉ាទ្រីស b ដែលមានចំនួនជួរដេក row ហើយចំនួនជួរឈរ col។
- គណនាផលបូកម៉ាទ្រីស៖

```
for(i = 0; i < row; i++){
    for(j = 0; j <col; j++){
        c[i][j] = a[i][j] + b[i][j];
    }
}
```

លំហាត់៖ ចូរសរសេរប្រូក្រាមសំរាប់គណនាផលបូករវាងម៉ាទ្រីសពីរផ្សេងគ្នា ដោយយើងត្រូវបញ្ចូលតំលៃអោយធាតុទាំងអស់របស់ម៉ាទ្រីសទាំងពីរ។ ជាចុងក្រោយបង្ហាញម៉ាទ្រីសទាំងបី។

```
#include<stdio.h>
#include<conio.h>
#define   P  printf
#define   S  scanf
void  main(){
        int  i, j, row, col, a[10][10], b[10][10], c[10][10];
Back1:  clrscr();
        P("Input Number of Rows = "); S("%d", &row);
        if(row<1 || row>10) goto  Back1;
Back2:
        P("Input Number of Columns = "); S("%d", &col);
        if(col<1 || col>10) goto  Back2;
        P("Input All elements of Matrix A:\n");
        for(i = 0; i < row; i++){
                for(j = 0; j < col; j++){
                        P("Input  a[%d][%d] = ", i, j);
                        S("%d", &a[i][j]);
                }
        }
```

```
P("Input All elements of Matrix B:\n");
for(i = 0; i < row; i++){
        for(j = 0; j < col; j++){
                P("Input  b[%d][%d] = ", i, j);
                S("%d", &b[i][j]);
        }
}
for(i = 0; i < row; i++){
        for(j = 0; j < col; j++){
                c[i][j] = a[i][j] + b[i][j];
        }
}
P("Matrix  A:\n");
for(i = 0; i < row; i++){
        for(j = 0; j < col; j++){
                P("%d\t", a[i][j]);
        }
        P("\n");
}
```

```
        P("Matrix  B:\n");
        for(i = 0; i < row; i++){
                for(j = 0; j < col; j++){
                        P("%d\t", b[i][j]);
                }
                P("\n");
        }

        P("Matrix  C = Matrix A + Matrix B:\n");
        for(i = 0; i < row; i++){
                for(j = 0; j < col; j++){
                        P("%d\t", c[i][j]);
                }
                P("\n");
        }
        getch();
}
```

១.៣.៤. របៀបគណនាផលគុណរវាងម៉ាទ្រីស៖

យើងអាចគណនាផលគុណរវាងម៉ាទ្រីស a[10][10] ជាមួយនឹងម៉ាទ្រីស b[10][10] តាមទ្រង់ទ្រាយដូចខាងក្រោម៖

- បញ្ចូលតំលៃ m, n និង p ចាប់ពី១ដល់១០។
- បញ្ចូលតំលៃអោយម៉ាទ្រីស a ដែលមានចំនួនជួរដេក m ហើយចំនួនជួរឈរ n។
- បញ្ចូលតំលៃអោយម៉ាទ្រីស b ដែលមានចំនួនជួរដេក n ហើយចំនួនជួរឈរ p។
- គណនាផលគុណម៉ាទ្រីស៖

$$(m \overset{a}{x} n) * (n \overset{b}{x} p) = (m \overset{c}{x} p)$$

$$c_{ij} = \sum_{k=0}^{n-1} (a_{ik} * b_{kj})$$

ដែល $i = 0, 1, 2, \ldots, m\text{-}1$

$j = 0, 1, 2, \ldots, p\text{-}1$

```
for(i = 0; i < m; i++){
    for(j = 0; j < p; j++){
        c[i][j] = 0;
        for(k = 0; k < n; k++)
            c[i][j] = c[i][j] + a[i][k]*b[k][j];
    }
}
```

លំហាត់៖ ចូរសរសេរប្រូក្រាមសំរាប់គណនាផលគុណរវាងម៉ាទ្រីសពីរផ្សេងគ្នា ដោយយើងត្រូវបញ្ចូលតំលៃអោយធាតុទាំងអស់របស់ម៉ាទ្រីសទាំងពីរ។ ជាចុងក្រោយបង្ហាញម៉ាទ្រីសទាំងបី។

```
#include<stdio.h>
#include<conio.h>
#define   P  printf
#define   S  scanf
void  main(){
          int  i, j, m, n, p, a[10][10], b[10][10], c[10][10];
Back1:    clrscr();
          P("Input Number of Rows of Matrix A = "); S("%d", &m);
          if(m<1 || m>10) goto  Back1;
Back2:
          P("Input Number of Columns of Matrix A or Rows of Matrix B = ");
          S("%d", &n);
          if(n<1 || n>10) goto  Back2;
Back3:
          P("Input Number of Columns of Matrix B = ");
          S("%d", &p);
          if(p<1 || p>10) goto  Back3;
```

```
P("Input All elements of Matrix A:\n");
for(i = 0; i < m; i++){
        for(j = 0; j < n; j++){
                P("Input  a[%d][%d] = ", i, j);
                S("%d", &a[i][j]);
        }
}
P("Input All elements of Matrix B:\n");
for(i = 0; i < n; i++){
        for(j = 0; j < p; j++){
                P("Input  b[%d][%d] = ", i, j);
                S("%d", &b[i][j]);
        }
}
for(i = 0; i < m; i++){
        for(j = 0; j < p; j++){
                c[i][j] = 0;
                for(int k = 0; k < n; k++)
                        c[i][j] = c[i][j] + a[i][k]*b[k][j];
        }
}
```

```
        P("Matrix  A:\n");
        for(i = 0; i < m; i++){
                for(j = 0; j < n; j++){
                        P("%d\t", a[i][j]);
                }
                P("\n");
        }
        P("Matrix  B:\n");
        for(i = 0; i < n; i++){
                for(j = 0; j < p; j++){
                        P("%d\t", b[i][j]);
                }
                P("\n");
        }

        P("Matrix  C = Matrix A * Matrix B:\n");
        for(i = 0; i < m; i++){
                for(j = 0; j < p; j++){
                        P("%d\t", c[i][j]);
                }
                P("\n");
        }
        getch();
}
```

២. String (Array of Characters)៖

នៅក្នុង C Programming Language គ្មានប្រភេទទិន្នន័យជា String ឡើយ ដោយវាមានតែប្រភេទទិន្នន័យជា char សំរាប់ផ្ទុកអក្សរមួយតួ។ String គឺជាការតំរៀបនៃតួអក្សរជាច្រើន ហេតុនេះយើងត្រូវប្រកាស អថេរសំរាប់ផ្ទុក String មួយក្នុងទ្រង់ទ្រាយដូចខាងក្រោម៖

char stringname[NumberOfcharacters+1];

ឧទាហរណ៍៖

char stuname[20]; មានន័យថា stuname គឺជា string មួយដែលអាចផ្ទុកចំនួន**19** តួអក្សរ។

char sex[2]; មានន័យថា sex គឺជា string មួយដែលអាចផ្ទុកចំនួន១តួអក្សរ។

ចំណាំ៖ ប្រសិនជាយើងផ្តុកតំរៀបតួអក្សរ មានចំនួនតិចជាងចំនួនតួអក្សរដែលយើងបានកំណត់អោយ String នោះតំបន់ដែលនៅសល់ត្រូវផ្តុក '\0' (Null Character)។

ឧទាហរណ៍៖ char st[8]= "Hello"; នោះការផ្តុករបស់វាត្រូវបង្ហាញដូចខាងក្រោម៖

H	e	l	l	o	\0	\0	\0

ក៏ប៉ុន្តែបើយើងផ្តុកឬកំណត់តួអក្សរអោយលើសចំនួនធាតុ នោះវានឹងមានភាព Error កើតឡើង។

ឧទាហរណ៍៖ char st[3]= "Hello"; នោះវានិង Error ព្រោះ Hello មានចំនួន៥តួអក្សរ ដែលលើសចំនួនអក្សររបស់ st ដែលមានត្រឹមតែពីរប៉ុណ្ណោះ។

២.១. ការបញ្ចូលនិងបញ្ចេញ String៖

+ យើងអាចប្រើ Statements ក្នុងឧទាហរណ៍មួយចំនួនដូចខាងក្រោម ដើម្បីបញ្ចូល String មួយ។

ឧទាហរណ៍៖

```
char st[30];
scanf("%s", &st);
```

ក្រោយដំណើរការ នោះ st ទទួលយក String ដែលនៅខាងមុខដកឃ្លាប៉ុណ្ណោះ។ ឧបមាយើងបញ្ចូល Welcome RUPP នោះ st ផ្ទុកពាក្យ "Welcome" ប៉ុណ្ណោះ។

ឧទាហរណ៍៖

```
char st[30];
scanf("%[\n]", &st);
```

ក្រោយដំណើរការនោះ st ទទួលយក String រហូតដល់ចុះបន្ទាត់ (\n) ទើបឈប់។ ឧបមាយើងបញ្ចូលពាក្យ Welcome RUPP រួចចុច Enter Key នោះ st នឹងផ្ទុកពាក្យ "Welcome RUPP"។

ឧទាហរណ៍៖

```
char st[30];
gets(st);
```

ក្រោយដំណើរការនោះ st ទទួលយក String រហូតដល់ចុះបន្ទាត់ (\n) ទើបឈប់។ ឧបមាយើងបញ្ចូលពាក្យ Welcome RUPP រួចចុច Enter Key នោះ st នឹងផ្ទុកពាក្យ "Welcome RUPP"។

+ យើងអាចប្រើ statements ក្នុងឧទាហរណ៍ដូចខាងក្រោម ដើម្បីបញ្ចេញ String មួយ។

ឧទាហរណ៍៖

puts(st); វានឹងបង្ហាញ String មួយ រួច cursor ស្ថិតនៅខាងដើមបន្ទាត់ថ្មី

printf("%s", st); វានឹងបង្ហាញ String មួយ រួច cursor ស្ថិតនៅខាងចុងនៃ String នោះ។

២.២. អនុគមន៍មួយចំនួនដែលត្រូវប្រើជាមួយនឹង String៖

យើងអាចប្រើអនុគមន៍មួយចំនួនជាមួយនឹង String បាន លុះត្រាតែយើងបានកំណត់ #include<string.h> ព្រោះដងខ្លួនរបស់អនុគមន៍ទាំងនោះស្ថិតនៅក្នុង Header file ឈ្មោះ string.h។ អនុគមន៍ទាំងនេះរួមមាន៖

+ strupr(st); មាននាទីបំប្លែង String ក្នុង st អោយក្លាយទៅជា Capital Letters។

+ strlwr(st); មាននាទីបំប្លែង String ក្នុង st អោយក្លាយទៅជា Small Letters។

+ strcmp(st1, st2); សំរាប់ប្រៀបធៀបរវាង string 2 គឺ st1 និង st2 ដោយ return តំលៃដូចខាងក្រោម៖

- បើ return តំលៃស្មើ 0 មានន័យថា st1 ដូចគ្នានឹង st2

- បើ return តំលៃធំជាង 0 មានន័យថា st1 ធំជាង st2

- បើ return តំលៃតូចជាង 0 មានន័យថា st1 តូចជាង st2

+ stricmp(st1, st2): មាននាទីដូច strcmp(st1, st2) ដែរ ដោយគ្រាន់តែវាមិនប្រកាន់ Capital Letter ឬ Small Letter។

+ strcpy(Dest, Source): មាននាទីចំលង String ពី Source ទៅអោយ Dest។

+ strstr(st1, st2): មាននាទីស្វែងរក string st2 នៅក្នុង st1។

ឧទាហរណ៍៖ ចូរសរសេរប្រូក្រាមមួយដើម្បីបញ្ចូល String រួចបញ្ចេញ String នោះ។ បន្ទាប់មកប្រើអនុគមន៍មួយចំនួនជាមួយនឹង String ដែលយើងបានបញ្ចូល។

```
#include <stdio.h>

#include <conio.h>

#include <string.h>

#define  P  printf

#define  S  scanf
```

```
void  main() {
        char st1[50], st2[50], st3[50];
        int n;
        P("Input First String:"); gets(st1);
        n = strlen(st1);
        P("Number of Characters in First String is %d\n", n);
        P(“Input Second String:”); gets(st2);
        if(strcmp(st1, st2)<0)
                P(“\“%s\”less than \“%s\”\n”, st1, st2);
        else if(strcmp(st1, st2)==0)
                P(“\“%s\”equal \“%s\”\n”, st1, st2);
        else
                P(“\“%s\”greater than \“%s\”\n”, st1, st2);
```

```
        P("Convert First String to Capital Letters: %s\n", strupr(st1));
        P("Convert First String to Small Letters: %s\n", strlwr(st1));
        strcpy(st3, st1);
        P("Copy First String to Third String: %s\n", st3);
        P("Press any key to exit..."); getch();
}
```

មេរៀនទី៦៖

Pointer

១. លក្ខណៈទូទៅរបស់ Pointer៖

១.១. និយមន័យ៖

Pointer គឺជាអថេរមួយប្រភេទដែលមានទាទីផ្ទុកលេខ Address នៃតំបន់ចាប់ផ្តើមរបស់អថេរមួយផ្សេងទៀត។

គេប្រើ Pointer ដើម្បីធ្វើអោយកម្មវិធីដំណើរការបានលឿនហើយចំណេញផ្ទៃនៃ RAM (Random Access Memory) ព្រោះ Pointer ត្រូវបានគេប្រើជំនួស Array សំរាប់កំណត់តំបន់របស់ RAM ដើម្បីផ្ទុកតំលៃនៃធាតុនីមួយៗរបស់ Array។

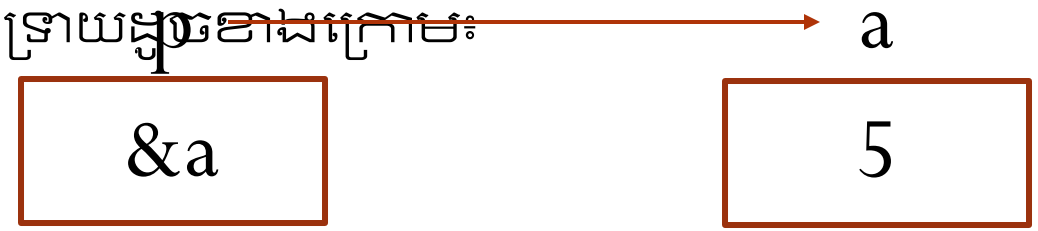
១.២. ការប្រកាសអថេរជា Pointer៖

ដើម្បីប្រកាសអថេរមួយដែលមានទាទីជា Pointer នោះយើងត្រូវសរសេរក្នុងទំរង់ដូចខាងក្រោម៖

Datatype *PointerName;

ឧទាហរណ៍: int *p; មានន័យថាអថេរ p មាននាទីជា pointer សំរាប់ផ្ទុកលេខ Address នៃតំបន់ចាប់ផ្តើមរបស់អថេរណាមួយ ដែលមានប្រភេទទិន្នន័យជា int។

ដើម្បីអោយ p ចង្អុលទៅកាន់អថេរ a មួយដែលមានប្រភេទទិន្នន័យជា int យើងត្រូវសរសេរក្នុងទ្រង់ទ្រាយដូចខាងក្រោម៖




```
int  a, *p;
a = 5;
p = &a;
```

ម្យ៉ាងវិញទៀតយើងអាចកំណត់តំលៃអោយអថេរ a តាមរយៈ p ដោយកំណត់ តំលៃអោយ *p (គឺសំដៅចំពោះអថេរដែល p កំពុងចង្អុល)។

ឧទាហរណ៍៖

```
int  a, *p;
p = &a;
*p = 3;          មានន័យថា a = 3
```

ឧទាហរណ៍៖ ចូរសរសេរកម្មវិធីដោយប្រើ pointer ដើម្បីរក Minimum និង Maximum រវាងពីចំនួនដែលត្រូវបញ្ចូល។

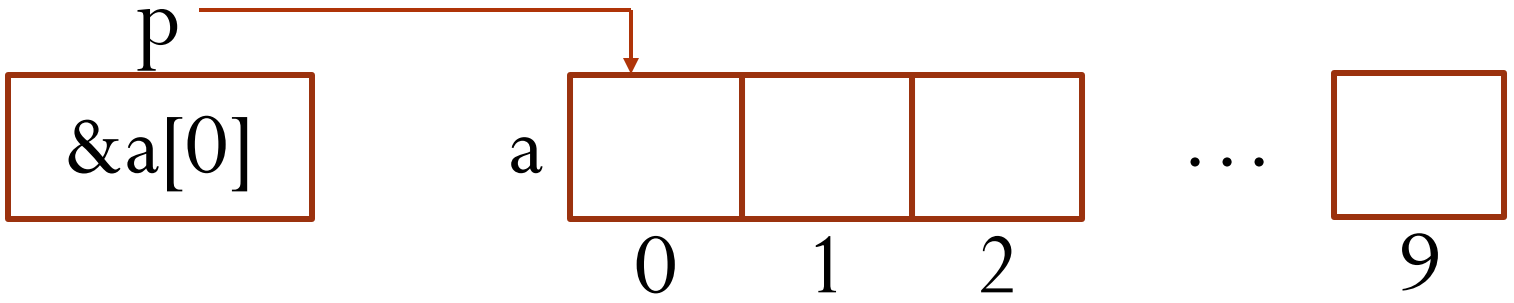
```
#include<stdio.h>
#include<conio.h>
#define  P   printf
#define  S   scanf
void  main(){
        int  a, b, c, *pa, *pb, *pc, min, max;
        pa = &a; pb = &b; pc = &c;
        clrscr();
        P("Input a="); S("%d", pa);
        P("Input b="); S("%d", pb);
        P("Input c="); S("%d", pc);
```

```
    if(*pa > *pb){
        min = *pb;   max = *pa;
    } else {
        min = *pa;   max = *pb;
    }
    if(min > *pc) min = *pc;
    if(max < *pc) max = *pc;
    P("Minimum(%d,%d,%d)=%d\n", *pa, *pb, *pc, min);
    P("Maximum(%d,%d,%d)=%d\n",*pa, *pb, *pc, max);
    P("Press any key to exit…");
    getch();
}
```

២. ការប្រើប្រាស់ Pointer ជំនួស Array មួយវិមាត្រៈ

ជាទូទៅ Pointer និង Array គឺមានទំនាក់ទំនងគ្នា ពីព្រោះឈ្មោះរបស់ Array មួយគឺជា Pointer Constant ដែលកំណត់ចង្អុលទៅកាន់ធាតុទី១ របស់ Array នោះ។

ឧបមាយើងប្រកាស int a[10]; នោះមានន័យថា a គឺជា Pointer Constant ដែលផ្ទុកនូវលេខ Address នៃតំបន់ចាប់ផ្តើមរបស់ a[0] ក៏ប៉ុន្តែ a មិនអាចប្រែប្រួលតំលៃបានឡើយ។ ឧបមាយើងប្រកាស int *p; ហើយកំណត់ p = a; ឬ p = &a[0]; នោះ p គឺជា Pointer ដែលចង្អុលទៅកាន់ Array a។



បើ p ផ្ទុក Address នៃ a[0] នោះ

p+1 ផ្ទុក Address នៃ a[1]

p+2 ផ្ទុក Address នៃ a[2]

.....

p+i ផ្ទុក Address នៃ a[i]

.....

p+n-1 ផ្ទុក Address នៃ a[n-1]

ហើយតំលៃនៃធាតុនីមួយៗត្រូវបានកំណត់ដូចខាងក្រោម៖

*p គឺជាតំលៃរបស់ a[0]

*(p+1) គឺជាតំលៃរបស់ a[1]

.....

*(p+i) គឺជាតំលៃរបស់ a[i]

.....

*(p+n-1) គឺជាតំលៃរបស់ a[n-1]

ចំណាំ៖ យើងអាចតំរៀបតំលៃរបស់ Array មួយវិមាត្រតាមលំដាប់កើន ដោយប្រើ Pointer តាមទ្រង់ទ្រាយដូចខាងក្រោម៖

```
int  j, temp;
for(i=0; i<n-1; i++)
        for(j=i+1; j<n; j++)
                if(*(p+i)>*(p+j)){
                        temp = *(p+i);
                        *(p+i) = *(p+j);
                        *(p+j) = temp;
                }
```

ឧទាហរណ៍៖ ចូរសរសេរកម្មវិធីមួយដោយប្រើ Pointer ដើម្បីបញ្ចូលតំលៃនៃ n ធាតុ របស់ Array មួយ (n ត្រូវបញ្ចូលតំលៃចាប់ពី២ឡើងទៅ) រួចបង្ហាញតំលៃនៃធាតុទាំងនោះ។ បន្ទាប់មករកតំលៃតូចបំផុត និងធំបំផុតរវាងចំនួនទាំងនោះ ទើបបង្ហាញតំលៃទាំងពីរនេះ។ បន្ទាប់មកតំរៀបតាមលំដាប់កើន ទើបបង្ហាញតំលៃនៃគ្រប់ធាតុក្រោយតំរៀបរួច។

```
#include<stdio.h>
#include<conio.h>
#define  P  printf
#define  S  scanf
void main(){
        int  i, n, a[100], *pa, min, max;
Back: clrscr();
        P("Input Number of Elements = "); S("%d", &n);
        if(n<2 || n>100) goto Back;
```

```
pa = a;
for(i=0; i<n; i++){
        P("Input a[%d] = ", i);
        S("%d", pa+i);
}
P("All elements of Array are:\n");
for(i=0; i<n; i++)
        P("%d\t", *(pa+i));
min = *pa; max = *pa;
for(i=1; i<n; i++){
        if(min>*(pa+i)) min = *(pa+i);
        if(max<*(pa+i)) max = *(pa+i);
}
P("\nMinimum = %d\n", min);
P("Maximum = %d\n", max);
```

```
    int  j, temp;
    for(i=0; i<n-1; i++)
        for(j=i+1; j<n; j++)
            if(*(pa+i)>*(pa+j)){
                temp = *(pa+i);
                *(pa+i) = *(pa+j);
                *(pa+j) = temp;
            }
    P("All elements of Array after sorting by Ascending are:\n");
    for(i=0; i<n; i++)
            P("%d\t", *(pa+i));
    P("\nPress any key to exit..."); getch();
}
```

ចំណាំ៖ យើងអាចប្រើអនុគមន៍មួយចំនួន ដើម្បីបង្កើតតំបន់នៃ RAM តាម រយៈ Pointer សំរាប់ផ្ទុកតំលៃរបស់ Array មួយវិមាត្រ ទៅតាមចំនួនធាតុរបស់ Array ដែលយើងបានកំណត់។ អនុគមន៍ទាំងនេះមានទំរង់ដូចខាងក្រោម៖

pointername=(Datatype *)malloc(n * sizeof(DataType));

ឬ

pointername=(Datatype *)calloc(n, sizeof(DataType));

ដោយដងខ្លួនរបស់អនុគមន៍ទាំងពីរនេះស្ថិតនៅក្នុង malloc.h ហេតុនេះ យើងត្រូវកំណត់ #include<malloc.h>។

ឧទាហរណ៍៖

pa = (int *)malloc(n*sizeof(int));

ឧទាហរណ៍៖ ចូរសរសេរប្រូក្រាមដើម្បីបញ្ចូលនិងបង្ហាញតំលៃនៃគ្រប់ធាតុរបស់ Array មួយវិមាត្រដែលមាន n ធាតុ (n ត្រូវបញ្ចូលតំលៃចាប់ពី២ឡើងទៅ) តាម រយៈ Pointer ក្នុងការកំណត់តំបន់របស់ RAM។

```
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
#define  P  printf
#define  S  scanf
void main(){
        int  i, n, *pa;
Back: clrscr();
        P("Input Number of Elements = "); S("%d", &n);
        if(n<2 ) goto Back;
        pa = (int *)malloc(n * sizeof(int));
```

```
        for(i=0; i<n; i++){
                P("Input a[%d] = ", i);
                S("%d", pa+i);
        }
        P("All elements of Array are:\n");
        for(i=0; i<n; i++)
                P("%d\t", *(pa+i));
        P("\nPress any key to exit..."); getch();
}
```

៣. ការប្រើប្រាស់ Pointer ជំនួស Array ពីរវិមាត្រ៖

យើងអាចប្រើ Pointer កំរិត២ (**pointerName) សំរាប់បង្កើតតំបន់របស់ RAM ដើម្បីផ្ទុកតំលៃនៃគ្រប់ធាតុរបស់ Array ២វិមាត្រ ដោយអនុវត្តតាមទ្រង់ទ្រាយដូចខាងក្រោម៖

```
int  i, j, row, col, **pa;
```

បញ្ចូលចំនួនជួរដេក (row) និងចំនួនជួរឈរ (col)

```
pa = (int **)malloc(row*sizeof(int));
for(i=0; i<row; i++)
        pa[i] = (int *)malloc(col*sizeof(int));
```

ឧទាហរណ៍៖ ចូរសរសេរប្រូក្រាមដោយប្រើ Pointer សំរាប់បញ្ចូលតំលៃអោយធាតុនីមួយៗរបស់ម៉ាទ្រីស A និង ម៉ាទ្រីស B។ បន្ទាប់មកកំណត់ម៉ាទ្រីស C ស្មើនឹងម៉ាទ្រីស A ដក ម៉ាទ្រីស B។ ជាចុងក្រោយបង្ហាញតំលៃរបស់ម៉ាទ្រីសទាំងបី។

```
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
#define  P  printf
#define  S  scanf
void  main(){
        int  i, j, row, col, **pa, **pb, **pc;
Back1:  clrscr();
        P("Input Number of Rows = "); S("%d", &row);
        if(row<2 ) goto Back1;
```

```
Back2:  P("Input Number of Columns = "); S("%d", &col);
        if(col<2 ) goto Back2;
        pa = (int **)malloc(row*sizeof(int));
        for(i=0; i<row; i++)
                pa[i] = (int *)malloc(col*sizeof(int));
        for(i=0; i<row; i++){
                for(j=0; j<col; j++){
                        P("Input  A[%d][%d] = ", i, j);
                        S("%d", &pa[i][j]);
                }
        }
```

```
pb = (int **)malloc(row*sizeof(int));
for(i=0; i<row; i++)
        pb[i] = (int *)malloc(col*sizeof(int));
for(i=0; i<row; i++){
        for(j=0; j<col; j++){
                P("Input  B[%d][%d] = ", i, j);
                S("%d", &pb[i][j]);
        }
}
```

```
pc = (int **)malloc(row*sizeof(int));
for(i=0; i<row; i++)
        pc[i] = (int *)malloc(col*sizeof(int));
for(i=0; i<row; i++)
        for(j=0; j<col; j++)
                pc[i][j] = pa[i][j] – pb[i][j];
P("Matrix A:\n");
for(i=0; i<row; i++){
        for(j=0; j<col; j++)
            P(“A[%d][%d] = %d\t", i, j, pa[i][j]);
        P("\n");
}
```

```
        P("Matrix B:\n");
        for(i=0; i<row; i++){
                for(j=0; j<col; j++)
                    P(“B[%d][%d] = %d\t", i, j, pb[i][j]);
                P("\n");
        }
        P("Matrix C = Matrix A – Matrix B:\n");
        for(i=0; i<row; i++){
                for(j=0; j<col; j++)
                    P(“C[%d][%d] = %d\t", i, j, pc[i][j]);
                P("\n");
        }
        P("Press any key to exit…"); getch();
}
```

មេរៀនទី៧៖

Structure

១. និយមន័យ៖

Structure គឺជាការប្រមូលផ្តុំនូវធាតុទាំងឡាយដែលមានប្រភេទទិន្នន័យផ្សេងពីគ្នាឬក៏ដូចគ្នា ក៏ប៉ុន្តែវាមានទំនាក់ទំនងគ្នា សំរាប់ផ្តុកនូវព័ត៌មានរបស់មនុស្សម្នាក់ វត្ថុអ្វីមួយ ទីកន្លែងណាមួយ ឬព្រឹត្តិការណ៍ណាមួយ។

ធាតុនីមួយៗនៅក្នុង Structure ត្រូវបានគេហៅថា Field ដែលមាននាទីផ្តុកនូវតំលៃទៅតាមប្រភេទទិន្នន័យរបស់វា។ តំលៃរបស់ Field ទាំងនោះផ្តុំគ្នាក្នុងមួយជួរដេកគឺបញ្ជាក់នូវព័ត៌មានជាក់លាក់ណាមួយ។

ដើម្បីប្រកាស Structure នោះយើងត្រូវអនុវត្តតាមទំរង់ដូចខាងក្រោម៖

```
struct  structureName {
        Datatype        FieldName1;
        …
        Datatype        FieldNameN;
```

ឧទាហរណ៍ទី១៖ យើងអាចកំណត់ Structure មួយសំរាប់ផ្ទុកទូវ
ព័ត៌មានរបស់និស្សិតម្នាក់ដែលមានអត្តលេខ (stuid), ឈ្មោះ
(stuname), ភេទ (gender) លេខទូរស័ព្ទ (phone) និងអាស័យដ្ឋាន
ទំនាក់ទំនង (contactAddress) តាមទ្រង់ទ្រាយដូចខាងក្រោម៖

```
struct  Student {
        long  stuid;
        char  stuname[30];
        char  gender[7];
        char  phone[20];
        char  contactAddress[200];
};
struct  Student  std;
```

ឧទាហរណ៍ទី២៖ យើងអាចកំណត់ Structure សំរាប់ផ្ទុកនូវព័ត៌មានរបស់សាស្ត្រាចារ្យម្នាក់ដែលមានលេខសំគាល់ (lecid), ឈ្មោះ (lecname), លេខទូរស័ព្ទ (phone), អាស័យដ្ឋានទំនាក់ទំនង (contactAddress) និងប្រាក់កំរៃក្នុងមួយម៉ោង (rate) តាមទ្រង់ទ្រាយដូចខាងក្រោម៖

```
struct  Lecturer {
        int  lecid;
        char  lecname[30];
        char  phone[20];
        char  contactAddress[200];
        float  rate;
};
struct  Lecturer  lec;
```

២. ដំណើរប្រតិបត្តិជាមួយ Field នីមួយៗរបស់ Structure៖

ដើម្បីដំណើរការជាមួយនឹង Field នីមួយៗរបស់ Structure យើងត្រូវអនុវត្តតាមទំរង់ដូចខាងក្រោម៖

structureVariable.FieldName

ឧទាហរណ៍ទី១៖ តាមរយៈ Structure ឈ្មោះ Student ក្នុងឧទាហរណ៍ទី១ នោះការប្រើប្រាស់ Field នីមួយៗត្រូវបានសរសេរដូចខាងក្រោម៖

- ចំពោះ stuid នោះយើងត្រូវសរសេរ std.stuid
- ចំពោះ stuname នោះយើងត្រូវសរសេរ std.stuname
- ចំពោះ gender នោះយើងត្រូវសរសេរ std.gender
- ចំពោះ phone នោះយើងត្រូវសរសេរ std.phone
- ចំពោះ contactAddress នោះយើងត្រូវសរសេរ std.contactAddress

ឧទាហរណ៍ទី២៖ តាមរយៈ Structure ឈ្មោះ Lecturer ក្នុងឧទាហរណ៍ទី ២ នោះការប្រើប្រាស់ Field នីមួយៗត្រូវបានសរសេរដូចខាងក្រោម៖

- ចំពោះ lecid នោះយើងត្រូវសរសេរ lec.lecid
- ចំពោះ lecname នោះយើងត្រូវសរសេរ lec.lecname
- ចំពោះ phone នោះយើងត្រូវសរសេរ lec.phone
- ចំពោះ contactAddress នោះយើងត្រូវសរសេរ lec.contactAddress
- ចំពោះ rate នោះយើងត្រូវសរសេរ lec.rate

៣. ការកំណត់តំលៃអោយ Field នីមួយៗរបស់ Structure៖

ដើម្បីកំនត់តំលៃឲ្យ Field ណាមួយរបស់ Structure នោះយើងត្រូវអនុវត្តតាមទំរង់ដូចខាងក្រោម៖

+ ចំពោះ String៖

strcpy(structureVariable.Fieldname, “string”);

ឧទាហរណ៍៖ strcpy(lec.lecname, “Kim Davy”);

strcpy(lec.phone, “012843728”);

+ ចំពោះប្រភេទទិន្នន័យផ្សេងៗទៀត៖

structureVariable.Fieldname = Value;

ឧទាហរណ៍៖ lec.lecid = 1;

lec.rate = 12.0;

៤. ការបញ្ចូលនិងបញ្ចេញនៃ Field នីមួយៗរបស់ Structure៖

ដើម្បីបញ្ចូលតំលៃឬបញ្ចេញតំលៃរបស់ Field នៃ Structure យើងត្រូវអនុវត្តដូចទៅនឹងការអនុវត្តន៍សំរាប់អថេរធម្មតាដែរ។ ក៏ប៉ុន្តែបើយើងចង់បញ្ចូលតំលៃអោយ Field ដែលមានប្រភេទទិន្នន័យជាចំនួនទសភាគ (float ឬ double) នោះយើងត្រូវបញ្ចូលតំលៃអោយអថេរធម្មតាណាមួយជាមុនសិន ទើបយើងផ្ទេរតំលៃពីអថេរធម្មតាទៅអោយ Field នោះ។

ឧទាហរណ៍៖ តាមរយៈ Structure ឈ្មោះ Lecturer ដែលបានប្រកាសក្នុងឧទាហរណ៍ទី២ខាងលើ នោះយើងអាចបញ្ចូលនិងបញ្ចេញតំលៃរបស់ Field នីមួយៗរបស់វាដូចខាងក្រោម៖

- ការបញ្ចូលតំលៃអោយ Field នីមួយៗ៖

  - ចំពោះ lecid នោះយើងត្រូវប្រើអនុគមន៍ដូចខាងក្រោម៖

    scanf("%d", &lec.lecid);

  - ចំពោះ lecname នោះយើងត្រូវប្រើអនុគមន៍ដូចខាងក្រោម៖

    gets(lec.lecname);

  - ចំពោះ phone នោះយើងត្រូវប្រើអនុគមន៍ដូចខាងក្រោម៖

    gets(lec.phone);

  - ចំពោះ rate នោះយើងត្រូវអនុវត្តដូចខាងក្រោម៖

    float r;

    scanf("%f", &r);

    lec.rate = r;

- ការបញ្ចេញតំលៃរបស់ Field នីមួយៗ៖

```
printf(“Lecturer’s Information:\n”);
printf(“Lecturer’s ID: %03d\n”, lec.lecid);
printf(“Lecturer’s Name: %s\n”, lec.lecname);
printf(“Phone: %s\n”, lec.phone);
printf(“Contact Address: %s\n”, lec.contactAddress);
printf(“Rate: $%0.2f\n”, lec.rate);
```

៥. Array នៃ Structure៖

យើងប្រើ Array នៃ Structure ដើម្បីផ្ទុកព័ត៌មានបានច្រើន ហើយព័ត៌មានទាំងនោះត្រូវបានគេហៅថា Records។ ដែល Record នីមួយៗកើតឡើងពីការផ្គុំរវាងតំលៃ Fields ទាំងអស់ក្នុងមួយជួរដេក។

ឧទាហរណ៍៖ Records ដែលបញ្ជាក់នូវព័ត៌មានរបស់និស្សិត

stuid	stuname	gender	phone	contactAddress
1	Chan Tola	M	0963728373	Phnom Penh
2	Long Thyda	F	0963718193	Kandal
…	…	…	…	…

ដើម្បីកំនត់ Array នៃ Structure នោះយើងត្រូវកំណត់តាមទំរង់ដូចខាងក្រោម៖

```
struct structureName structureVariable[Number of Records];
```

ឧទាហរណ៍៖ តាមរយៈ Structure ឈ្មោះ Student ដែលយើងបានប្រកាសរួចហើយ នោះយើងអាចកំនត់ Array នៃ Structure ដូចខាងក្រោម៖

```
struct Student std[100];
```

ហេតុដូចនេះដើម្បីសរសេរទាក់ទងទៅនឹង Field នីមួយៗរបស់វា យើងត្រូវកំនត់ដូចខាងក្រោម៖

- ចំពោះ Field ឈ្មោះ stuid យើងត្រូវសរសេរ std[i].stuid
- ចំពោះ Field ឈ្មោះ stuname យើងត្រូវសរសេរ std[i].stuname
- ចំពោះ Field ឈ្មោះ gender យើងត្រូវសរសេរ std[i].gender
- ចំពោះ Field ឈ្មោះ phone យើងត្រូវសរសេរ std[i].phone
- ចំពោះ Field ឈ្មោះ contactAddress យើងត្រូវសរសេរ std[i].contactAddress

ដែល i មានតំលៃចាប់ពី ០ ដល់ n-1 (ដែល n គឺជាចំនួន Records ទាំងអស់)។

ឧទាហរណ៍៖ ចូរសរសេរប្រូក្រាមសំរាប់បញ្ចូលព័ត៌មានរបស់និស្សិតចំនួន n នាក់ (ដែល n ត្រូវបញ្ចូលតំលៃចាប់ពី២ដល់១០០)។ ចំពោះនិស្សិតម្នាក់ៗរួមមានលេខសំគាល់ (stuid) ត្រូវមានតំលៃចាប់ពី១ហើយកើន១ជាស្វ័យប្រវត្តិ ចំណែកឯឈ្មោះ (stuname) ភេទ (gender) និងមធ្យមភាគ (average) យើងត្រូវបញ្ចូលតំលៃ។ ជាចុងក្រោយបង្ហាញព័ត៌មានរបស់និស្សិតទាំងអស់នោះ ក្នុងលក្ខណៈជាតារាង។

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#define P  printf
#define S  scanf
void main(){
      struct Student {
            int stuid;
            char stuname[30];
            char gender[7];
            float average;
      };
      struct Student std[100];
```

```
        int  i, n;
        float  avg;
Back: clrscr();
        P("Input Number of Students: "); S("%d", &n);
        if(n<2 || n>100) goto  Back;
        for(i=0; i<n; i++){
                std[i].stuid = i + 1;
                fflush(stdin);
                P("Input Student's Name: ");
                gets(std[i].stuname);
                P("Input Student's Gender: ");
                gets(std[i].gender);
                P("Input Average: ");  S("%f", &avg);
                std[i].average = avg;
        }
```

```
        P("Student's Information:\n");
        P("ID:\tName:\t\tGender:\tAverage:\n");
        P("-------------------------------------------------------\n");
        for(i=0; i<n; i++)
                P("%02d\t%s\t\t%s\t%0.2f\n", std[i].stuid,
                std[i].stuname, std[i].gender, std[i].average);
        P("-------------------------------------------------------\n");
        P("Press any key to exit…"); getch();
}
```

+ ដើម្បីតំរៀបតាមលំដាប់កើននៃឈ្មោះរបស់និស្សិត (Sorting Student's Name By Ascending) នោះយើងត្រូវសរសេរក្នុងទ្រង់ទ្រាយដូចខាងក្រោម៖

```
int  j;
struct Student  temp;
for(i=0; i<n-1; i++)
   for(j=i+1; j<n; j++)
      if(strcmpi(std[i].stuname, std[j].stuname)>0){
                  temp = std[i]; std[i] = std[j]; std[j] = temp;
      }
```

ម៉្យាងទៀតបើយើងចង់តំរៀបតាមលំដាប់ចុះនៃតំលៃមធ្យមភាគ (Sorting Average By Descending) យើងគ្រាន់តែដាក់លក្ខខណ្ឌដូចខាងក្រោម៖

លំហាត់៖ ចូរសរសេរប្រូក្រាមដោយប្រើ Structure សំរាប់បញ្ចូលព័ត៌មានរបស់សាស្ត្រាចារ្យចំនួន n នាក់ (ដោយ n ត្រូវបញ្ចូលតំលៃចាប់ពី២ ដល់ ១០០)។ ចំពោះសាស្ត្រាចារ្យម្នាក់ៗត្រូវមាន លេខសំគាល់ (lecid), ឈ្មោះ (lecname), លេខទូរស័ព្ទ (phone), ប្រាក់កំរែក្នុងមួយម៉ោង (rate)និង ចំនួនម៉ោងបង្រៀន (hours) បន្ទាប់មកគណនារកប្រាក់ខែ (salary) ដោយយកចំនួនម៉ោងបង្រៀនគុណនឹងប្រាក់កំរែក្នុងមួយម៉ោង។ ជាចុងក្រោយតំរៀបតាមលំដាប់កើននៃឈ្មោះ រួចបង្ហាញព័ត៌មានរបស់សាស្ត្រាចារ្យទាំងអស់ក្នុងទំរង់ជាតារាង។

ចំណាំ៖ ចំពោះលេខសំគាល់ត្រូវចាប់ផ្តើមពី១ ហើយកើន១ជាស្វ័យប្រវត្តិ ចំណែកឯប្រាក់កំរែក្នុងមួយម៉ោង (rate) ចំនួនម៉ោងបង្រៀន (hours) និង ប្រាក់ខែ (salary) គឺជាចំនួនទសភាគ។

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#define  P   printf
#define  S   scanf

void  main(){
        struct  Lecturer {
                int  lecid;
                char  lecname[30];
                char  phone[20];
                float  hours, rate;
                float  salary;
        };
        struct  Lecturer  lec[100];
        int  i, j, n;
        float  h, r;
```

```
Back: clrscr();
        P("Input Number of Lecturers: "); S("%d", &n);
        if(n<2 || n>100) goto  Back;
        for(i=0; i<n; i++){
                lec[i].lecid = i + 1;
                fflush(stdin);
                P("Input Lecturer's Name: ");
                gets(lec[i].lecname);
                P("Input Lecturer's Phone: ");
                gets(lec[i].phone);
                P("Input Hours: ");  S("%f", &h);  lec[i].hours = h;
                P("Input Rate: ");  S("%f", &r);  lec[i].rate = r;
                lec[i].salary = h*r;
        }
```

```
    struct  Lecturer  temp;
    for(i=0; i<n-1; i++)
       for(j=i+1; j<n; j++)
              if(strcmpi(lec[i].lecname, lec[j].lecname)>0){
                     temp = lec[i]; lec[i] = lec[j]; lec[j] = temp;
              }
    P("Lecturer's Information:\n");
    P("ID:\t Name:\t\tPhone:\tHours\tRate\tSalary:\n");
    P("-------------------------------------------------------\n");
    for(i=0; i<n; i++)
              P("%03d\t%s\t\t%s\t%0.2f\t$%0.2f\t%$0.2f\n",
                     lec[i].lecid, lec[i].lecname, lec[i].phone,
                     lec[i].hours, lec[i].rate, lec[i].salary);
    P("-------------------------------------------------------\n");
    P("Press any key to exit…"); getch();
}
```

៦. ការប្រើ Pointer ជាមួយនឹង Structure៖

យើងអាចប្រើប្រាស់ Pointer សំរាប់ចង្អុលទៅកាន់ Structure ដោយកំណត់ Structure Variable គឺជា Pointer ហើយយើងអាចកំណត់តំបន់របស់ RAM សំរាប់ផ្ទុកនូវ Records ដោយប្រើអនុគមន៍ malloc តាមទ្រង់ទ្រាយដូចខាងក្រោម៖

Syntax៖

```
struct  structureName *pointerName;
```

+ កំណត់តំបន់របស់ RAM សំរាប់ផ្ទុក Records ចំនួន n៖

```
pointerName = (struct structureName*)
                malloc(n*sizeof (struct structureName));
```

ឧទាហរណ៍៖ តាមរយៈ Structure ដែលយើងបានប្រកាសក្នុងឧទាហរណ៍ខាងលើ យើងអាចកំណត់ Structure Variable ជា Pointer រួចកំណត់ Memory Allocation សំរាប់ផ្ទុកព័ត៌មានរបស់និស្សិតចំនួន n នាក់ ដូចខាងក្រោម៖

```
struct  Student  *pstd;
pstd = (struct  Student *) malloc (n*sizeof(struct Student));
```

ហេតុនេះដើម្បីដំណើរការ Field នីមួយៗរបស់ Structure ដោយប្រើ Pointer នោះយើងត្រូវកំណត់សរសេរតាមទំរង់ដូចខាងក្រោម៖

+ ចំពោះ stuid យើងត្រូវសរសេរក្នុងទំរង់ដូចខាងក្រោម៖

(*(pstd+i)).stuid; ឬ (pstd+i)->stuid

ដែល i=0, 1, …, n-1

+ ចំពោះ stuname យើងត្រូវសរសេរក្នុងទំរង់ដូចខាងក្រោម៖

(*(pstd+i)).stuname; ឬ (pstd+i)->stuname

ឧទាហរណ៍៖ ចូរសរសេរប្រូក្រាមសំរាប់បញ្ចូលព័ត៌មានរបស់និស្សិតចំនួន n នាក់ (ដែល n ត្រូវបញ្ចូលតំលៃចាប់ពី២ឡើងទៅ) ដោយប្រើប្រាស់ Pointer ក្នុងការកំណត់តំបន់របស់ RAM សំរាប់ផ្ទុកព័ត៌មានទាំងនោះ។ បន្ទាប់មកបង្ហាញព័ត៌មានរបស់និស្សិតទាំងអស់ ក្នុងលក្ខណៈជាតារាង។ ចំពោះព័ត៌មានរបស់និស្សិតម្នាក់ៗ រួមមានអត្តលេខ (stuid), ឈ្មោះ (stuname), ភេទ (gender), លេខទូរស័ព្ទ (phone) និងមធ្យមភាគ(average)។

```
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
#include<string.h>
#define  P   printf
#define  S   scanf
void  main(){
        struct  Student {
                int  stuid;
                char  stuname[30];
                char  gender[7];
                char  phone[20];
                float  average;
        };
        struct  Student  *pstd;
        int  i, n;
        float  avg;
```

```
Back: clrscr();
        P("Input Number of Students: "); S("%d", &n);
        if(n<2) goto  Back;
        pstd = (struct  Student *)malloc(n*sizeof(struct   Student));
        for(i=0; i<n; i++) {
                (pstd+i)->stuid = i + 1;
                fflush(stdin);
                P("Input Student's Name: ");
                gets((pstd+i)->stuname);
                P("Input Student's Gender: ");
                gets((pstd+i)->gender);
                P("Input Student's Phone: ");
                gets((pstd+i)->phone);
                P("Input Average: ");  S("%f", &avg);
                (pstd+i)->average = avg;
        }
```

```
        P("Student's Information:\n");
        P("ID:\tName:\t\tGender:\tPhone:\tAverage:\n");
        P("-------------------------------------------------------\n");
        for(i=0; i<n; i++)
                P("%03d\t%s\t\t%s\t%s\t%0.2f\n", (pstd+i)->stuid,
                (pstd+i)->stuname, (pstd+i)->gender,
                (pstd+i)->phone, (pstd+i)->average);
        P("-------------------------------------------------------\n");
        P("Press any key to exit..."); getch();
}
```

លំហាត់៖ ចូរសរសេរប្រូក្រាមសំរាប់បញ្ចូលព័ត៌មានរបស់សាស្ត្រាចារ្យចំនួន n នាក់ (ដែល n ត្រូវបញ្ចូលតំលៃចាប់ពី២ឡើងទៅ) ដោយប្រើប្រាស់ Pointer ក្នុងការកំណត់តំបន់របស់ RAM សំរាប់ផ្ទុកព័ត៌មានទាំងនោះ។ ចំពោះសាស្ត្រាចារ្យម្នាក់ៗត្រូវមាន៖ លេខសំគាល់ (lecid), ឈ្មោះ (lecname), លេខទូរស័ព្ទ (phone), ប្រាក់កំរែក្នុងមួយម៉ោង (rate)និងចំនួនម៉ោងបង្រៀន (hours) បន្ទាប់មកគណនារកប្រាក់ខែ (salary) ដោយយកចំនួនម៉ោងបង្រៀនគុណនឹងប្រាក់កំរែក្នុងមួយម៉ោង។ ជាចុងក្រោយតំរៀបតាមលំដាប់ចុះនៃប្រាក់ខែ រួចបង្ហាញព័ត៌មានរបស់សាស្ត្រាចារ្យទាំងអស់ក្នុងទំរង់ជាតារាង ព្រមទាំងបង្ហាញប្រាក់ខែសរុប។

ចំណាំ៖ ចំពោះលេខសំគាល់ត្រូវចាប់ផ្តើមពី១ ហើយកើន១ជាស្វ័យប្រវត្តិ ចំណែកឯប្រាក់កំរែក្នុងមួយម៉ោង (rate) ចំនួនម៉ោងបង្រៀន (hou

```
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
#define  P   printf
#define  S   scanf

void  main(){
        typedef  struct  Lecturer {
                int  lecid;
                char  lecname[30];
                char  phone[20];
                float  hours, rate, salary;
        };
        Lecturer  *plec;
        int  i, j, n;
        float  h, r, t=0;
```

```
Back: clrscr();
        P("Input Number of Lecturers: "); S("%d", &n);
        if(n<2) goto  Back;
        plec = (Lecturer *)malloc(n*sizeof(Lecturer));
        for(i=0; i<n; i++){
                (plec+i)->lecid = i + 1;
                fflush(stdin);
                P("Input Lecturer's Name: ");
                gets((plec+i)->lecname);
                P("Input Lecturer's Phone: ");
                gets((plec+i)->phone);
                P("Input Hours: ");  S("%f", &h); (plec+i)->hours = h;
                P("Input Rate: ");  S("%f", &r); (plec+i)->rate = r;
                (plec+i)->salary = h*r;
                t = t + h*r;
        }
```

```
Lecturer  temp;
for(i=0; i<n-1; i++)
   for(j=i+1; j<n; j++)
            if((plec+i)->salary< (plec+j)->salary){
              temp=*(plec+i);*(plec+i)=*(plec+j);*(plec+j)=temp;
            }
P("Lecturer's Information:\n");
P("ID:\t Name:\t\tPhone:\tHours\tRate\tSalary:\n");
P("-------------------------------------------------------\n");
for(i=0; i<n; i++)
    P("%03d\t%s\t\t%s\t%0.2f\t$%0.2f\t%$0.2f\n",
            (plec+i)->lecid,(plec+i)->lecname, (plec+i)->phone,
            (plec+i)->hours, (plec+i)->rate, (plec+i)->salary);
P("-------------------------------------------------------\n");
P("                                  Total Salary = $%0.2f\n", t);
P("Press any key to exit…"); getch();
}
```

មេរៀនទី៨៖

FILE

១. លក្ខណៈទូទៅ៖

File គឺជាការប្រមូលផ្តុំទិន្នន័យរក្សាទុកជានិរន្តរ៍នៅក្នុង Secondary Storage (Hard Disk)។ នៅក្នុង C Programming Language នោះ File មានពីរប្រភេទដូចជា៖

+ Text File៖ ទិន្នន័យដែលត្រូវផ្ទុកក្នុង Text File មានលក្ខណៈជា Graphical Characters ដែលត្រូវបានរៀបចំជាខ្សែបន្ទាត់ហើយនៅចុងនៃខ្សែបន្ទាត់នីមួយៗមាន ‘\n’ ។

+ Binary File៖ ទិន្នន័យដែលត្រូវផ្ទុកក្នុង Binary File មានលក្ខណៈជា Internal Data Format មានន័យថាវាអាចផ្ទុកនូវចំនួនគត់ ចំនួនទសភាគនិងតួអក្សរដោយមិនចាំបាច់បំប្លែងឡើយ។

២. អនុគមន៍សំរាប់បើកនិងបិទ File៖

២.១. អនុគមន៍សំរាប់បើក File៖

យើងអាចប្រើអនុគមន៍ fopen( ) សំរាប់បើក File មួយ ដើម្បីយកទិន្នន័យរបស់វាមកផ្ទុកលើ RAM ជាបណ្តោះអាសន្នក្នុងគោលបំណងមួយចំនួនទៅតាម Mode របស់វា។

Syntax៖

```
FILE *fp;
fp = fopen("filename.extension", "mode");
```

ដែល៖

- filename.extension គឺជាឈ្មោះនិងលក្ខណៈសំគាល់របស់ File ដែលត្រូវបើកឬបង្កើត។ ជាទូទៅ Text File មានលក្ខណៈសំគាល់ (extension) ".txt" ចំណែក binary file មានលក្ខណៈសំគាល់ ".bin"។

- mode គឺជារបៀបនៃការបើក File ដើម្បីធ្វើអ្វីមួយទៅតាមប្រភេទរបស់វាដូចខាងក្រោម៖

mode (text file)	អត្ថន័យ
r (read)	បើក file សំរាប់តែអានទិន្នន័យបុណ្ណោះ (read only)
w (write)	បើកឬបង្កើត file សំរាប់សរសេរទិន្នន័យថ្មីចូលហើយទិន្នន័យចាស់នឹងបាត់ទាំងអស់។
a (append)	បើកឬបង្កើត file សំរាប់សរសេរទិន្នន័យថ្មីចូលបន្ត
r+	បើក file សំរាប់អានឬសរសេរទិន្នន័យថ្មីចូល
w+	បើកឬបង្កើត file សំរាប់អានឬសរសេរទិន្នន័យថ្មីចូល
a+	បើកឬបង្កើត file សំរាប់អានឬសរសេរទិន្នន័យថ្មីចូល បន្ត។

mode (binary file)	អត្ថន័យ
rb	បើក file សំរាប់តែអានទិន្នន័យបុណ្ណោះ (read only)
wb	បើកឬបង្កើត file សំរាប់សរសេរទិន្នន័យថ្មីចូល ហើយទិន្នន័យចាស់នឹងបាត់ទាំងអស់។
ab	បើកឬបង្កើត file សំរាប់សរសេរទិន្នន័យថ្មីចូលបន្ត
r+b	បើក file សំរាប់អានឬសរសេរទិន្នន័យថ្មីចូល
w+b	បើកឬបង្កើត file សំរាប់អានឬសរសេរទិន្នន័យថ្មីចូល
a+b	បើកឬបង្កើត file សំរាប់អានឬសរសេរទិន្នន័យថ្មីចូល បន្ត។

២.២. អនុគមន៍សំរាប់បិទ File៖

សន្មត់ថាយើងបានប្រកាសនិងបើក File ដោយប្រើ fp នោះអនុគមន៍សំរាប់បិទ File មានទំរង់ដូចខាងក្រោម៖

Syntax៖ fclose(fp);

៣. អនុគមន៍សំរាប់សរសេរឬអានទិន្នន័យទាក់ទងនឹង Binary File៖

៣.១. អនុគមន៍សំរាប់សរសេរទិន្នន័យចូលក្នុង Binary File៖

ក. fwrite( ) ជាអនុគមន៍សំរាប់សរសេរទិន្នន័យចូលក្នុង Binary File។

Syntax៖ fwrite(&address, sizeof(datatype), count, filepointer);

ដែល៖

- address គឺជាទីតាំងនៃតំបន់ចាប់ផ្តើមរបស់អថេរមួយរឺ object ណាមួយដែលត្រូវផ្ទុកតំលៃទៅក្នុង Binary File។
- count គឺជាចំនួនដែលយើងត្រូវកំនត់ចាប់ពី១ឡើងទៅ ដើម្បីផ្ទុកតំលៃរឺ object មួយរឺច្រើនចាប់ពីទីតាំងចាប់ផ្តើម។
- filepointer គឺមាននាទីចង្អុលទៅកាន់ file stream ដែលយើងបានបើកដោយប្រើអនុគមន៍ fopen( ) ក្នុងទ្រង់ទ្រាយ Binary File។

ឧទាហរណ៏៖

ក. ចូរបង្កើត structure Student មួយដែលមាន fields ដូចខាងក្រោម៖

stuid, stuname, sex, average

ខ. ចូរសរសេរកម្មវិធីដើម្បីបញ្ចូលព័ត៌មានរបស់និស្សិត n នាក់

គ. ចូរសរសេរទិន្នន័យពី structure Student ទៅរក្សាទុកក្នុង Binary File មួយមាន ឈ្មោះថា student.bin។

ដំណោះស្រាយ៖

```
#include<stdio.h>
#include<conio.h>
void main(){
        clrscr();
        typedef struct Student{
                int stuid;
                char stuname[30], sex[7];
                float average;
        };
        Student stu[30];
```

```
    int i, n;
    float avg;
    FILE *fp;
    fp=fopen("Student.bin",  "ab");
    printf("Input Number of Student:"); scanf("%d", &n);
    for(i=0; i<n; i++){
            stu[i].stuid = i +1;
            printf("Input Name: "); fflush(stdin);
            gets(stu[i].stuname);
            printf("Input Sex: "); gets(stu[i].sex);
            printf("Input Average:"); scanf("%f", &avg);
            stu[i].average = avg;
            fwrite(&stu[i], sizeof(Student),  1, fp);                    }
    fclose(fp);          getch( );
}
```

៣.២. អនុគមន៍សំរាប់អានទិន្នន័យពីក្នុង Binary File៖

ក. fread( ) ជាអនុគមន៍សំរាប់អានទិន្នន័យចេញពីក្នុង Binary File។

Syntax៖ fread(&address, sizeof(datatype), count, filepointer);

ដែល៖

- address គឺជាទីតាំងនៃតំបន់ចាប់ផ្តើមរបស់អថេរមួយរឺ object ណាមួយ ដែលត្រូវផ្ទុកតំលៃទៅក្នុង Binary File។
- count គឺជាចំនួនដែលយើងត្រូវកំនត់ចាប់ពី១ឡើងទៅ ដើម្បីផ្ទុកតំលៃរឺ object មួយរឺច្រើនចាប់ពីទីតាំងចាប់ផ្តើម។
- filepointer គឺមាននាទីចង្អុលទៅកាន់ file stream ដែលយើងបានបើក ដោយប្រើអនុគមន៍ fopen( ) ក្នុងទ្រង់ទ្រាយ Binary File។

ខ. fseek( ) គឺជាអនុគមន៍សំរាប់ផ្លាស់ទីអោយ file pointer ចង្អុលត្រង់ទីតាំង ណា មួយនៃ file stream។

Syntax៖ fseek(filepointer, long offset, int origin);

ដែលៈ

- filepointer គឺមាននាទីចង្អុលទៅកាន់ file stream ដែលយើងបានបើក។
- offset: គឺជាចំនួន byte ទាំងឡាយគិតពីចំនុចគោលកំនត់ (origin) ដើម្បីកំនត់បាននូវទីតាំងសំគាល់ថ្មីរបស់ file stream។
- origin: មានតំលៃដូចជា 0, 1 & 2 ដោយក្នុងនោះ 0 (SEEK_SET) កំនត់ចំនុចចាប់ផ្តើម របស់ file ចំណែក ឯ 1 (SEEK_CUR) កំណត់ទីតាំងបច្ចុប្បន្ន ហើយ 2 (SEEK_END) កំណត់ទីតាំងចុងក្រោយរបស់ file។

ឧទាហរណ៍ៈ

- fseek(fp, 4, 0); មានន័យថាកំណត់ទីតាំងដែល file pointer ស្ថិតនៅគឺ 4bytes ពីចំណុចចាប់ផ្តើមរបស់ file stream។
- fseek(fp, -4, 2); មានន័យថាកំណត់ទីតាំងដែល file pointer ស្ថិតនៅគឺ 4bytes ខាងមុខចំណុចបញ្ចប់របស់ file stream។
- fseek(fp, 4, 1); មានន័យថាកំណត់ទីតាំងដែល file pointer ស្ថិតនៅគឺ 4bytes ពីចំណុចបច្ចុប្បន្ន របស់ file stream។
- fseek(fp, 0, 0); មានន័យថាកំណត់ទីតាំងត្រង់ចំណុចចាប់ផ្តើម របស់ file stream ដែល file pointer ចង្អុលទៅកាន់។ ហើយវាសម មូលនឹង rewind(fp);។

គ. rewind( ) គឺជាអនុគមន៍ដែលគេប្រើសំរាប់ចល័តទីតាំងរបស់ file pointer ត្រលប់ទៅកាន់ទីតាំងចាប់ផ្តើម។

Syntax៖ rewind(file pointer);

ឃ. ftell( ) គឺជាអនុគមន៍សំរាប់ប្រាប់ពីទីតាំងដែល file pointer ស្ថិតនៅគិតជា byte។

Syntax៖ ftell(file pointer);

ឧទាហរណ៍៖

ក. ចូរបង្កើត Structure Student មួយដែលមាន fields ដូចខាងក្រោម៖

stuid, stuname, sex, average

ខ. ចូរសរសេរកម្មវិធីដើម្បីអានទិន្នន័យចេញពី Binary File ឈ្មោះ student.bin យកមកក្សាទុកក្នុង Structure Student។

គ. ចូរសរសេរកម្មវិធីដើម្បីបង្ហាញលទ្ធផលមកលើអេក្រង់។

ដំណោះស្រាយ៖

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
void main(){
        clrscr();
        typedef struct Student{
                int stuid;
                char stuname[30], sex[7];
                float average;
        };
        Student stu[30];
        FILE *fp;
        fp=fopen("Student.bin", "rb");
        int i, n;
```

```
    if(fp==NULL){
        printf("File Not Found...");
        getch();
        exit(0);
    }
    fseek(fp, 0, 2);
    n=ftell(fp)/sizeof(Student);
    rewind(fp);  //fseek(fp, 0, 0);
    fread(stu, sizeof(Student), n, fp);
    printf("ID:\tName:\t\tSex:\tAverage:\n");
    for(i=0; i<n; i++){
        printf("%d\t", stu[i].stuid);
        printf("%s\t", stu[i].stuname);
        printf("%s\t", stu[i].sex);
        printf("%0.2f\n", stu[i].average);
    }
    fclose(fp);
    getch( );
}
```

៤. អនុគមន៍សំរាប់សរសេរទិន្នន័យឬអានទិន្នន័យទាក់ទងជាមួយនឹង Text File៖

៤.១. អនុគមន៍សំរាប់សរសេរទិន្នន័យចូលក្នុង Text File៖

សន្មត់ថាយើងបានប្រកាសនិងបើក Text File ដោយប្រើ fp នោះអនុគមន៍សំរាប់សរសេរទិន្នន័យចូលក្នុង Text File មួយដូចខាងក្រោម៖

+ fputc( ) គឺជាអនុគមន៍ដែលមាននាទីសំរាប់សរសេរតួអក្សរមួយតួទៅផ្ទុកក្នុង File stream ដែលត្រូវបានចង្អុលដោយ fp។

Syntax៖fputc(ch, fp);

ដែល ch គឺជាអថេរដែលមានប្រភេទជា char ដោយយើងត្រូវបញ្ចូលអក្សរមួយតួ។

ឧទាហរណ៍៖ ចូរសរសេរកម្មវិធីមួយដើម្បីបង្កើត File មួយឈ្មោះ letter.txt ដើម្បីសរសេរតួអក្សរទាំងឡាយរហូតដល់ជួបសញ្ញា ! ទើបឈប់។

```
#include<stdio.h>
#include<conio.h>
void main(){
    clrscr();
    FILE *fp;
    char ch;
    fp=fopen("letter.txt", "w");
    printf("Please input string (Press ! to stop):");
    while(ch=getchar(), ch!='!'){
        fputc(ch, fp);
    }
    fclose(fp);
}
```

+ fprintf( ) ជាអនុគមន៍ដែលមាននាទីអានតំលៃចេញពីអថេរជាប្រើនទៅផ្ទុកក្នុង Text File ដោយតំលៃទាំងនោះត្រូវបានបំលែងទៅជា Graphical character។

Syntax៖

fprintf(fp, “format control”, arg1,…..,argN);

ឧទាហរណ៍៖ ចូរសរសេរកម្មវិធីមួយដើម្បីកត់ត្រានូវព័ត៌មានរបស់បុគ្គលិក n នាក់ (n ត្រូវបញ្ជូលតំលៃចាប់ពី២ឡើងទៅ) ក្នុង Text file មួយឈ្មោះ employee.txt ដែលព័ត៌មានទាំងអស់នោះរួមមាន emp_id, emp_name, emp_gender, salary។

```c
#include<stdio.h>
#include<conio.h>
void main(){
    clrscr();
    FILE *fp;
    fp = fopen("employee.txt", "a");
    int i, n;
    int emp_id;
    char emp_name[30], emp_gender[7];
    float salary;
    printf("Please Input Number of Employee:");
    scanf("%d", &n);
    for(i=0; i<n; i++){
        printf("Input ID:"); scanf("%d", &emp_id);
        printf("    Name:"); fflush(stdin);
        gets(emp_name);
        printf("  Gender:"); fflush(stdin);
        gets(emp_gender);
        printf("  Salary:"); scanf("%f", &salary);
        fprintf(fp, "%d\t%s\t%s\t%.2f\n",
                emp_id, emp_name, emp_gender, salary);
    }
    fclose(fp);
}
```

៤.២. អនុគមន៍សំរាប់អានទិន្នន័យពីក្នុង Text File៖

សន្មត់ថាយើងបានប្រកាស fp គឺជា File pointer សំរាប់ចង្អុលទៅកាន់ File stream ដែលត្រូវបានបើក នោះអនុគមន៍សំរាប់អានទិន្នន័យមានដូចជា៖

+ fgetc( ) គឺជាអនុគមន៍សំរាប់អានអក្សរមួយតួចេញពី File stream មកផ្ទុកក្នុងអថេរដែលយើងបានកំណត់អោយមានប្រភេទទិន្នន័យជា char។

Syntax៖

```
char  ch;
ch=fgetc(fp);
```

ឧទាហរណ៍: ចូរសរសេរកម្មវិធីដើម្បីអានតួអក្សរទាំងអស់ពីក្នុង Text file មួយឈ្មោះ letter.txt មកបង្ហាញលើអេក្រង់។

```c
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
void main(){
    clrscr();
    FILE *fp;
    fp = fopen("letter.txt", "r");
    char ch;
    if(fp==NULL){
        printf("File Not Found...");
        getch();
        exit(0);
    }
    while((ch=fgetc(fp))!=EOF)
        printf("%c", ch);
    fclose(fp);
}
```

+ fscanf( ) គឺជាអនុគមន៍សំរាប់អានទិន្នន័យចេញពី file stream របស់ text file មកផ្ទុកក្នុងអថេរមួយឬច្រើន។

Syntax៖

fscanf(fp, “Format Control”, &Variable1,…, &VariableN);

ឧទាហរណ៍: ចូរសរសេរកម្មវិធីដើម្បីអានទិន្នន័យចេញពីក្នុង text file មួយឈ្មោះ employee.txt យកមកផ្ទុកក្នុងអថេរ emp_id, emp_name, emp_gender, salary រួចបង្ហាញមកលើអេក្រង់។

```
#include<stdio.h>
#include<conio.h>
#define P printf
#define S scanf
#define FS fscanf
void main(){
        clrscr();
        FILE *fp;
        int   emp_id;
        char   emp_name[30], emp_gender[7];
        float   salary;
        fp = fopen("employee.txt", "r");
        while((FS(fp, "%d%*c%[^\t]%*c%[^\t]%f%*c",
          &emp_id, emp_name, emp_gender, &salary))!=EOF){
                P("%d\t%s\t%s\t%0.2f\n",emp_id, emp_name,
                        emp_gender, salary);
        }
        getch();
        fclose(fp);
}
```