บทที่ 9 คลาส

9.1 บทนำ

Introduction

ในการศึกษาจากบทเรียนที่แล้วท่านได้ใช้โครงสร้างในการสร้างชนิดข้อมูลแบบรวมไว้ ซึ่งประกอบด้วยเซ็ตของดาต้าเมมเบอร์ ในการศึกษาบทเรียนนี้ท่านจะได้ขยายแนวความคิดของการสร้างชนิดข้อมูลแบบรวมด้วยการผูกดาต้าเมมเบอร์และฟังก์ชันที่จะใช้ในการจัดการเข้าไว้ด้วยกันเป็นหนึ่งเดียวโดยการใช้คลาสของภาษา C++ คลาสเป็นพื้นฐานในการเขียนโปรแกรมเชิงวัตถุ (object-oriented programming) และเป็นตัวที่ทำให้ภาษา C++ นี้แตกต่างจากภาษาที่ใช้เขียนโปรแกรมแบบเก่าโดยทั่วๆไป เช่น ภาษา C, Pascal, และ FORTRAN เป็นต้น

ในหัวข้อที่ 9.2 ท่านจะได้ศึกษาถึงวิธีในการประกาศและนิยามคลาส วิธีการสร้างตัวแปรคลาส (ออบเจ็ก-objects) วิธีการสร้างเมมเบอร์ฟังก์ชันในการเข้าถึงข้อมูลที่เก็บอยู่ในดาต้าเมมเบอร์ของออบเจ็ก ในหัวข้อที่ 9.3 ท่านจะได้สร้างเมมเบอร์ฟังก์ชันในการแก้ไขออบเจ็ก และในหัวข้อที่ 9.4 ท่านจะได้สร้างฟังก์ชันในการแสดงออบเจ็กสองชนิดบนจอภาพ คือ จุดและหน้าต่างข้อความ

9.1 คลาสคืออะไร

What Is A Class?

ชนิดข้อมูล Point ที่แสดงดังข้างล่างนี้ประกอบด้วยคู่ของเลขจำนวนเต็มซึ่งระบุพิกัด x และ y บนพื้นระนาบ

```
class Point
{  …
   int x,      // x-coordinate
       y;      // y-coordinate
};
```

ในการใช้ตัวแปรชนิด Point ท่านต้องการฟังก์ชันในการทำงานพื้นฐาน เช่น เซ็ตค่าพิกัดเริ่มต้นของจุด ย้ายจุด แสดงจุด และอื่นๆ ท่านสามารถสร้างฟังก์ชัน เรียกว่า movePoint(), displayPoint(), และอื่นๆก็ได้ แต่วิธีการที่มีประสิทธิภาพมากกว่านั้นก็คือ การรวมเอาฟังก์ชันต่างๆดังกล่าวนั้นให้เป็นส่วนหนึ่งของชนิดข้อมูล Point ผลลัพธ์ที่ได้ก็คือ คลาสจะประกอบด้วยชุดของดาต้าเมมเบอร์และชุดของฟังก์ชัน เรียกว่า เมมเบอร์ฟังก์ชันซึ่งคอยจัดการกับดาต้าเมมเบอร์เหล่านั้น ในการประกาศคลาสต่อไปนี้จากไฟล์ point.h จะประกอบด้วยเมมเบอร์ฟังก์ชันหลายฟังก์ชันซึ่งทำงานกับจุดในพื้นระนาบ

```
// Declaration for a two-dimensional Point class.
class Point
{
 public:        // Member functions

   // Constructors
   Point ( int xCoord, int yCoord );   // Define a point

   // Derived point attribute
   double distance ();                 // Distance from (0,0)

   // Modification functions
   void move ( int dx, int dy );       // Move point
   void exchangeXY ();                 // Exchange coordinates

   // Point display function
   void display ();

   // Outputs the data members -- used in testing and debugging
   void showDataMembers ();

 private:      // Data members
```

```
    int x,      // x-coordinate
        y;      // y-coordinate
};
```

ท่านประกาศตัวแปร (หรือออบเจ็ก) ชนิด Point ได้ดังต่อไปนี้

```
Point pt (2 , 5) ;
```

ในการประกาศนี้เป็นการสร้างออบเจ็กซึ่งมีพิกัด x คือ 2 และพิกัด y คือ 5 จากบทที่ 8 ท่านได้ใช้โอเปอเรเตอร์จุด ในการเข้าถึงดาต้าเมมเบอร์ของโครงสร้าง ในกรณีของคลาสท่านใช้โอเปอเรเตอร์จุดในการเข้าถึงดาต้าเมมเบอร์ของออบเจ็กและในการเรียกใช้เมมเบอร์ฟังก์ชัน ตัวอย่างเช่น ในการแสดงจุด pt ท่านเรียกใช้โดยการใช้ชื่อออบเจ็กตามด้วยโอเปอเรเตอร์จุดและตามด้วยเมมเบอร์ฟังก์ชัน

```
pt.display();
```

นอกจากนี้ยังมีอีกหลายเรื่องที่จะต้องกล่าวถึงในการประกาศคลาส Point ในการประกาศจะประกอบด้วยส่วนที่เป็น public ซึ่งประกอบด้วยต้นแบบของเมมเบอร์ฟังก์ของคลาส และ ส่วนที่เป็น private ซึ่งประกอบด้วยดาต้าเมมเบอร์ ในการประกาศดาต้าเมมเบอร์ให้อยู่ในส่วนที่เป็น private เป็นการป้องกันไม่ให้ฟังก์ชันที่ไม่ใช่สมาชิกของคลาส Point เข้าถึงและจัดการดาต้าเมมเบอร์ของคลาส Point ได้โดยตรง ดังโปรแกรมต่อไปนี้

```
void main ()
{
    point pt (1 , 2) ;
    pt.x = 5 ;                  // Error – attempt to access private data member
}
```

จะเกิดข้อผิดพลาดขึ้นในขณะทำการคอมไพล์ เพราะว่า x คือ ไพรเวทดาต้าเมมเบอร์ (private data member) และสามารถเข้าถึงได้โดยผ่านทางเมมเบอร์ฟังก์ชันใดฟังก์ชันหนึ่งของคลาส Point เท่านั้น หรืออีกนัยหนึ่งก็คือ ท่านสามารถเข้าถึงไพรเวทดาต้าเมมเบอร์ได้ทางอ้อมโดยการใช้เมมเบอร์ฟังก์ชันของคลาส Point นั่นเอง ซึ่งฟังก์ชันต่างๆเหล่านี้อยู่ในส่วนของ public และสามารถ

เรียกใช้ผ่านทางฟังก์ชันไหนๆก็ได้ ไม่ว่าฟังก์ชันนั้นจะเป็นสมาชิกหรือไม่ใช่สมาชิกก็ตาม สรุปโดยรวมก็คือ เมมเบอร์ฟังก์ชันเป็นส่วนที่ใช้ในการติดต่อกับคลาส Point

ในการประกาศคลาส Point จะแสดงรายการของดาต้าเมมเบอร์ของคลาสและเมมเบอร์ฟังก์ชัน ในการจะใช้คลาส Point ท่านจะต้องทำการสร้างเมมเบอร์ฟังก์ชันขึ้นมาเสียก่อน เราลองมาดูถึงวิธีการในการสร้างเมมเบอร์ฟังก์ชัน ในแต่ละฟังก์ชันเราจะกำหนดเงื่อนไขหรือความต้องการ (preconditions or requirements) ที่จะต้องมีก่อนที่ฟังก์ชันจะถูกเรียก และ ผลลัพธ์ของฟังก์ชันนั้น ซึ่งก็คือค่าข้อมูลที่ฟังก์ชันจะส่งกลับออกมาหรือสิ่งที่ฟังก์ชันนั้นจะกระทำ รูปแบบดังกล่าวนี้จะใช้ในการกำหนดคุณลักษณะของฟังก์ชัน

Point (int xCoord, int yCoord)

Requirements

None

Results

Constructor. Creates a point with coordinates xCoord and yCoord.

double distance ()

Requirements

None

Results

Returns the distance from a point to the origin (0,0).

ฟังก์ชันแรกนี้เป็นฟังก์ชันพิเศษเรียกว่า คอนสตรักเตอร์(constructor) คอนสตรักเตอร์จะจองเนื้อที่ในหน่วยความจำสำหรับออบเจ็กและเซ็ตค่าเริ่มต้นของดาต้าเมมเบอร์ คอนสตรักเตอร์ของคลาสใดคลาสหนึ่งจะถูกเรียกใช้โดยอัตโนมัติ เมื่อมีการประกาศออบเจ็กในโปรแกรม ตัวอย่างเช่น การประกาศ

Point nextPt (2 , 4) ;

จะเป็นการเรียกคอนสตรักเตอร์ของคลาส Point ซึ่งเป็นการจองเนื้อที่สำหรับออบเจ็ก nextPt และเซ็ตค่าเริ่มต้นดาต้าเมมเบอร์ x และ y ของออบเจ็ก nextPt ให้เท่ากับ 2 และ 4 ตามลำดับ ให้

สังเกตว่าคอนสตรักเตอร์จะมีชื่อเดียวกันกับชื่อคลาสที่มันอยู่ เช่นในกรณีนี้ ชื่อ Point และไม่มีชนิดข้อมูลที่ส่งกลับ ในการสร้างคอนสตรักเตอร์ของคลาส Point แสดงอยู่ด้านล่าง สำหรับ scope resolution operator (::) ใช้ในการบอกคอมไพเลอร์ว่าฟังก์ชันนี้เป็นเมมเบอร์ฟังก์ชันของคลาส Point

```
Point::Point (int xCoord, int yCoord )
{
    x = xCoord;
    y = yCoord;
}
```

ในฟังก์ชันที่สองของคลาส Point คือ ฟังก์ชัน distance() ส่งค่าข้อมูลระยะทางของจุดจากจุดเริ่มต้น (0,0)

```
double Point::distance ( )
// Returns the distance from a point to the origin (0,0).
{
    return ( sqrt ( pow(x,2) + pow(y,2) ) ) ;
}
```

ในทำนองเดียวกัน scope resolution operator ใช้ในการระบุว่า ฟังก์ชันนี้เป็นส่วนหนึ่งของคลาส Point ฟังก์ชัน distance() สามารถอ้างอิงถึงคาต้าเมมเบอร์ x และ y ได้เพราะว่าฟังก์ชันนี้เป็นเมมเบอร์ฟังก์ชันของคลาส Point นั่นเอง ให้สังเกตว่าท่านไม่ต้องมีการประกาศพารามิเตอร์ x และ y ในฟังก์ชัน distance()

ตัวอย่างโปรแกรมต่อไปนี้แสดงการเรียกใช้ฟังก์ชัน distance()

```
void main ()
{
        Point alpha(1,2),
                Beta(3,5);
```

```
        cout  << alpha.distance() << endl;
        cout << beta.distance() << endl;
}
```

เมื่อมีการเรียก alpha.distance() ฟังก์ชัน distance() จะมีการเรียกใช้ดาต้าเมมเบอร์ x และ y ของออบเจ็ก alpha ในขณะที่มีการเรียก beta.distance ก็จะมีการเรียกใช้ดาต้าเมมเบอร์ของออบเจ็ก beta แทน

โดยปกติแล้วในส่วนของการประกาศคลาส และในส่วนของการสร้างเมมเบอร์ฟังก์ชันนั้น มักจะแยกเก็บไว้คนละไฟล์กัน ในส่วนของการประกาศคลาสจะเก็บไว้ในเฮดเดอร์ไฟล์ภายใต้ชื่อที่เหมาะสม เช่นในกรณีของคลาส Point จะเก็บไว้ในเฮดเดอร์ไฟล์ชื่อ point.h และในส่วนของการสร้างจะเก็บไว้โดยใช้ชื่อเดียวกันแต่นามสกุลต่างกัน รายละเอียดของการสร้างเมมเบอร์ฟังก์ชันของคลาส Point ซึ่งเก็บอยู่ในไฟล์ point.cpp จะแสดงอยู่ด้านล่าง การประกาศต้นแบบคลาสในไฟล์ point.h และส่วนการสร้างเมมเบอร์ฟังก์ชันในไฟล์ point.cpp รวมกันถือว่าเป็นการประกาศคลาส Point นั่นเอง

```
#include <iostream.h>
#include <math.h>
#include "point.h"

//------------------------------------------------------------------

Point::Point ( int xCoord, int yCoord )

// Constructor.

{
   x = xCoord;
   y = yCoord;
}

//------------------------------------------------------------------

double Point::distance ()
```

```
// Returns the distance from a point to the origin (0,0).

{
   return ( sqrt( pow(x,2) + pow (y,2) ) );
}

//---------------------------------------------------------------------

void Point::showDataMembers ()

// Displays a point's data members. Use for testing/debugging only.

{
   cout << "(" << x << " " << y << ")";
}
```

ให้สังเกตว่าไฟล์ไหนก็ตามที่มีการอ้างอิงถึงคลาส Point จะต้องมีการรวมเฮดเดอร์ไฟล์ point.h เอาไว้ด้วยเพื่อที่คอมไพเลอร์จะได้ใช้ในการอ้างอิงได้

การทดลอง

ฟังก์ชัน main() จากไฟล์ testpt.cpp ต่อไปนี้ใช้คลาส Point ในการคำนวณระยะทางจากจุดสองจุดว่าห่างจากจุดเริ่มต้นเท่าไร

```
// Test program for the Point class

#include <iostream.h>
#include "point.h"

void main()
{
   // Construct the points (10,5) and (7,8).
```

```
Point pt1(10,5),
      pt2(7,8);

// Display both point's data members.
cout << endl << "Point 1: " ;
pt1.showDataMembers();
cout << endl << "Point 2: " ;
pt2.showDataMembers();

// Display the distance from each point to (0,0).
cout << endl << "Distance from Point 1 to the origin: "
    << pt1.distance();
cout << endl << "Distance from Point 2 to the origin: "
    << pt2.distance() << endl;

// Remove the comment specifier (///) from the code below to test
// the move(), exchangeXY(), and display() functions.

// Move pt1's x-coordinate by 3 and its y-coordinate by -2.
// pt1.move(3,-2);

// Display pt1's new coordinates.
// cout << "Point 1: " ;
// pt1.showDataMembers();
// cout << endl;

// Exchange pt1's x- and y-coordinates.
// pt1.exchangeXY();

// Display pt1's new coordinates.
```

```
    // cout << "Point 1: " ;
    // pt1.showDataMembers();
    // cout << endl;

    // Display an asterisk on the screen at pt1's coordinates.
    // pt1.display();
}
```

โปรแกรมนี้จะประกอบขึ้นมาจากไฟล์ 3 ไฟล์ คือ testpt.cpp (เก็บฟังก์ชัน main() ไว้) ไฟล์ point.h (เก็บส่วนของการประกาศคลาส Point) และไฟล์ point.cpp (เก็บส่วนของการสร้างเมมเบอร์ฟังก์ชันของคลาส Point)

ขั้นตอนที่ 1 ให้ตรวจสอบกับคู่มือของคอมไพเลอร์ว่าในการสร้างโปรแกรมจากหลายๆไฟล์จะทำได้อย่างไร
ขั้นตอนที่ 2 ให้ทำการคอมไพล์ส่วนของการสร้างคลาส Point จากไฟล์ point.cpp
ขั้นตอนที่ 3 ให้ทำการคอมไพล์โปรแกรมจากไฟล์ testpt.cpp
ขั้นตอนที่ 4 ให้ทำการลิงค์ออบเจ็กไฟล์จากขั้นตอนที่ 2 และ 3
ขั้นตอนที่ 5 ให้ทำการเอ็กซิคิ้วท์โปรแกรมที่ได้จากขั้นตอนที่ 4

การทดลอง

วินโดวส์ (windows) เป็นเนื้อที่ส่วนพิเศษที่วาดขึ้นมาบนจอภาพ ในการทดลองนี้ท่านจะได้ทำการสร้างคลาสขึ้นมาเพื่อที่จะใช้ในการสร้าง การเปลี่ยนขนาด การย้ายตำแหน่ง และการแสดงวินโดวส์ที่มีข้อความข้างใน การประกาศสำหรับคลาส TextWindow จากไฟล์ twindow.h มีรายละเอียดดังต่อไปนี้

```
// Declarations for the TextWindow class.

const int MAX_TEXT_LENGTH = 101;   // Maximum length of a text string

class TextWindow
{
```

```
public:

  // Constructor
  TextWindow ( int leftEdge, int topEdge,
               int rightEdge, int bottomEdge,
               char textString[]              );

  // Derived window attributes
  int width ();                          // Window width
  int height ();                         // Window height

  // Window modification functions
  void move ( int dx, int dy );             // Move a window
  void resize ( int dx, int dy );           // Resize a window
  void replaceText ( char textString[] );   // Replace window text

  // Outputs the data members -- used in testing/debugging
  void showDataMembers ();

  // Window display function
  void display ();

private:

  // Data members
  int left,                    // Positions of the window's edges
      top,
      right,
      bottom;
  char text[MAX_TEXT_LENGTH];   // Window's text string
```

};

ขั้นตอนที่ 1 ให้ทำการสร้างชุดของเมมเบอร์ฟังก์ชันต่อไปนี้

TextWindow (int leftEdge, int topEdge, int rightEdge,

Int bottomEdge, char textString[ ]) ;

Requirements

None

Results

Constructor. Create a window that is bounded by the specified edges and contains textString.

int width ()

Requirements

None

Results

Returns the width of a window.

int height ()

Requirements

None

Results

Returns the height of a window.

ขั้นตอนที่ 2 ให้เก็บส่วนของการสร้างเมมเบอร์ฟังก์ชันเหล่านี้ไว้ในไฟล์ twindow.cpp

ขั้นตอนที่ 3 ให้เพิ่มส่วนของการสร้างเมมเบอร์ฟังก์ชัน showDataMembers () ซึ่งเก็บอยู่ในไฟล์ showdata.cpp เข้าไปในไฟล์ twindow.cpp

ขั้นตอนที่ 4 ให้เติมแผนการทดสอบต่อไปนี้ให้สมบูรณ์ สำหรับคอนสตรักเตอร์ ฟังก์ชัน width() และ ฟังก์ชัน height()

กรณีทดสอบ	ตัวอย่างข้อมูล	ผลลัพธ์ที่คาดการณ์	ตรวจสอบ
Construct a 40x10 (40 wide		top =	

by 10 high) text window containning the text string “Sample” Construct a 20x5 text window containing the text string “My text”		left = right = bottom = text = Sample width = 40 height = 10 top = left = right = bottom = text = My text width = 20 height = 5	

9.3 การเปลี่ยนแปลงออบเจ็ก

Modifying An Object

ถึงแม้ว่าผู้ใช้จะไม่สามารถเข้าถึงดาต้าเมมเบอร์ของออบเจ็กในส่วน private ได้โดยตรง แต่ท่านสามารถสร้างเมมเบอร์ฟังก์ชันในส่วน public เพื่อที่จะให้ผู้ใช้สามารถเข้าถึง(เพื่อดึงข้อมูลหรือแก้ไข)ดาต้าเมมเบอร์ของออบเจ็กโดยทางอ้อมก็ได้ ตัวอย่างเมมเบอร์ฟังก์ชันต่อไปนี้จะทำการย้ายจุดไปตามทิศทาง x และ y ที่กำหนด ซึ่งก็เป็นการแก้ไขดาต้าเมมเบอร์ x และ y ของจุดนั่นเอง

```
void Point::move (int dx, int dy)
// Moves a point by the specified amounts.
{
    x += dx;
    y += dy;
}
```

ส่วนของโปรแกรมต่อไปนี้จะทำการย้ายจุดจากตำแหน่งพิกัด (12,4) ไปที่พิกัด (8,7)

```
// Construct a point with an initial x-coordinate of 12 and an
```

```
// initial y-coordinate of 4.
Point pt(12,4) ;

// Move pt's x-coordinate by –4 and pt's y-coordinate by +3
pt.move(-4,3) ;
```

ให้สังเกตว่าดาต้าเมมเบอร์ของออบเจ็ก pt จะถูกแก้ไขโดยปราศจากการอ้างอิงถึงโดยตรง

การทดลอง

เมมเบอร์ฟังก์ชันต่อไปนี้จะทำการเปลี่ยนตำแหน่งพิกัด x และ y ของจุด

```
void exchangeXY ()
Requirements
None
Results
Exchanges a point's x- and y-coordinate.
```

ให้เติมโปรแกรมภาษา C++ ต่อไปนี้ให้สมบูรณ โครงร่างของโปรแกรมนี้เก็บอยู่ในไฟล์ point.cpp

```
void _____ ::exchangeXY ()
{

}
```

การทดลอง

ในการทดลองนี้ท่านจะได้เพิ่มเมมเบอร์ฟังก์ชันเข้าไปที่คลาส TextWindow เมมเบอร์ฟังก์ชันแต่ละตัวนั้นจะทำการแก้ไขดาต้าเมมเบอร์ของ TextWindows ในลักษณะต่างๆกัน

```
void move (int dx, int dy)
```

Requirements

None

Results

Moves a window by the specified amounts.

Void resize (int dx, int dy)

Requirements

None

Results

Resizes a window by moving its lower, right-hand corner by the specified amounts.

Void replaceText (char textString [ ])

Requirements

None

Results

Replaces a windows' text with textString

ขั้นตอนที่ 1 ให้ทำการสร้างเมมเบอร์ฟังก์ชันที่ระบุไว้ด้านบน ต้นแบบของฟังก์ชันเหล่านี้จะอยู่ในส่วนประกาศของคลาส TextWindow ซึ่งเก็บอยู่ในไฟล์ twindow.h

ขั้นตอนที่ 2 ให้ใส่ฟังก์ชันต่างๆเหล่านี้เข้าไปในส่วนของการสร้างคลาส TextWindow ซึ่งเก็บอยู่ในไฟล์ twindow.cpp

ขั้นตอนที่ 3 ให้เติมแผนการทดสอบสำหรับฟังก์ชัน move() ต่อไปนี้

กรณีทดสอบ	ตัวอย่างข้อมูล	ผลลัพธ์ที่คาดการณ์	ตรวจสอบ
Construct a 40x10 (40 wide by 10 high) text window containning the text string "Sample"		top = left = right = bottom = text = Sample width = 40 height = 10	

Move the window constructed above –5 units horizontally and –2 units vertically	10 2	top = left = right = bottom = text = Sample width = 40 height = 10	
Move the window constructed above –5 units horizontally and –2 units vertically	-5 -2	top = left = right = bottom = text = Sample width = 40 height = 10	

ขั้นตอนที่ 4 ให้เติมแผนการทดสอบสำหรับฟังก์ชัน resize() ต่อไปนี้

กรณีทดสอบ	ตัวอย่างข้อมูล	ผลลัพธ์ที่คาดการณ์	ตรวจสอบ
Construct a 40x10 (40 wide by 10 high) text window containning the text string “Sample”		top = left = right = bottom = text = Sample width = 40 height = 10	
Make the window larger by 10 units horizontally and 2 units vertically	10 2	top = left = right = bottom = text = Sample width = 50	

		height = 12	
Make the window larger by 10 units horizontally and 2 units vertically	-5 -3	top = left = right = bottom = text = Sample width = 35 height = 7	

ขั้นตอนที่ 5

กรณีทดสอบ	ตัวอย่างข้อมูล	ผลลัพธ์ที่คาดการณ์	ตรวจสอบ
Construct a 40x10 (40 wide by 10 high) text window containning the text string “Sample”		top = left = right = bottom = text = Sample width = 40 height = 10	
Replace the text string with “New Word”	New word	top = left = right = bottom = text = Sample width = 40 height = 10	
Replace the text string with your name		top = left = right = bottom = text = Sample width = 40 height = 10	

9.4 การแสดงออบเจ็ก

Displaying An Object

ในหัวข้อนี้ท่านจะได้สร้างฟังก์ชันซึ่งแสดงจุดและวินโดวส์ข้อความไปที่จอภาพ ในสภาพแวดล้อมของการเขียนโปรแกรมทั่วๆไปจะมีฟังก์ชันซึ่งช่วยให้ท่านสามารถเปลี่ยนตำแหน่งของเคอร์เซอร์ไปยังตำแหน่งที่ต้องการบนจอภาพ และแสดงข้อความเริ่มต้นจากที่ตำแหน่งนั้น ก่อนที่จะเริ่มต้นทดลองในหัวข้อนี้ท่านจะต้องทำการสำรวจว่าในระบบที่ท่านใช้อยู่มีฟังก์ชันอะไรที่ทำหน้าที่ดังกล่าวและระบบพิกัดจอภาพของท่านเป็นอย่างไร เช่น จุดพิกัดของมุมทั้ง 4 จุดของจอภาพคืออะไร

การทดลอง

เมมเบอร์ฟังก์ชันของคลาสพอยน์ต่อไปนี้ จะแสดงเครื่องหมายดาว ณ ตำแหน่งพิกัดของจุดที่กำหนด

void display()

Requirements

None

Results

Display an asterisk on the screen at a point's coordinates.

ให้เติมฟังก์ชันของภาษา C++ ต่อไปนี้ให้สมบูรณ์ โครงร่างของฟังก์ชันนี้เก็บอยู่ในไฟล์ point.cpp

```
void _____ :: display ()
// Displays a point as an asterisk (*) on the screen.
{



}
```

การทดลอง

เมมเบอร์ฟังก์ชันของ TextWindow ต่อไปนี้จะแสดงวินโดวส์ข้อความบนจอภาพ

void display ()

Requirements

The window's edges lie within the screen boundaries.

Results

Displays a text window. Begins by displaying the window's border and then displays the window's text string within the border. Treats vertical bars (|) in the windows's text string as end-of-line markers and crops the text to fit the window's borders.

จอภาพซึ่งแสดงวินโดวส์ข้อความ 3 วินโดวส์ดังแสดงข้างล่าง ในแต่ละวินโดวส์จะมีข้อความแสดงอยู่ให้สังเกตว่าวินโดวส์ข้อความด้านล่างทั้ง 2 วินโดวส์จะตัดข้อความบางส่วนที่เกินกว่าที่จะแสดงได้ออก

ขั้นตอนที่ 1 ให้ทำการสร้างเมมเบอร์ฟังก์ชัน display() ที่ระบุไว้ข้างต้น ต้นแบบของฟังก์ชันนี้เก็บอยู่ในส่วนประกาศของคลาส TextWindow ซึ่งเก็บอยู่ในไฟล์ twindow.h

ขั้นตอนที่ 2 ให้เพิ่มฟังก์ชันนี้เข้าไปในส่วนของการสร้างของคลาส TextWindow ซึ่งเก็บอยู่ในไฟล์ twindow.cpp

ขั้นตอนที่ 3 ให้เติมแผนการทดสอบต่อไปนี้ให้สมบูรณ์

กรณีทดสอบ	ตัวอย่างข้อมูล	ผลลัพธ์ที่คาดการณ์	ตรวจสอบ
Construct a 40x10 text window containing the text string “Sample” Display a text window containing a multiline text string Display a text window containing a cropped text string			