ชื่อ-นามสกุล ..... รหัสประจำตัวนักศึกษา .....

01076104 Programming Project

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

การทดลองที่ 8 : พื้นฐานการเขียน Text-mode Game ครั้งที่ 4

จุดประสงค์

นักศึกษาสามารถพัฒนาเกมในส่วนของการกำหนดกรอบการแสดงผล การใช้ Event ต่างๆ เพื่อใช้ Mouse และ Keyboard รวมถึงเทคนิคการวาดรูปในหน้าจอแบบ Double Buffer

ข้อกำหนด

- โปรแกรมมีการพัฒนาและทดสอบบน Visual C++ 2019 Edition การปฏิบัติการโดยใช้ compiler ตัวอื่นๆ จะต้องมีการแก้ไขโปรแกรม

ตอนที่ 1 การกำหนดขนาดของจอภาพและ buffer ส่วนแสดงผล

ในการใช้งานระบบ windows โดยทั่วไป ผู้ใช้งานสามารถกำหนดขนาดหน้าจอ console และการแสดงผลอื่นๆ ได้ โดยคลิ๊กขวาที่แถบด้านบนของหน้าจอ cmd -> properties -> Layout โดยจะมี 2 ค่าที่ผู้ใช้งานสามารถปรับเปลี่ยนได้คือ

- windows size สำหรับแก้ไขความกว้าง (width : จำนวนตัวอักษรต่อบรรทัด) และความยาว (height:จำนวนบรรทัดที่แสดงผล)
- screen buffer size สำหรับกำหนดขนาดหน่วยความจำในการเก็บค่าข้อมูลที่ได้แสดงผลไปแล้ว โดยสามารถตั้งค่า Width และ Heigh ได้เหมือนกับ Windows Size ในกรณีที่ค่า ScreenBufferSize:Height มากกว่า WindowsSize:Height จะทำให้มี ScrollBar ด้านขวาเพื่อให้ Scroll ดูข้อมูลก่อนหน้าได้ และในลักษณะเดียวกันเมื่อ ScreenBufferSize:Width มากกว่า WindowsSize:Height จะทำให้มี ScrollBar ด้านล่างเพื่อให้สามารถ Scroll ดูข้อมูลได้เช่นกัน

ในการเขียนโปรแกรมเพื่อปรับแต่งค่าการแสดงผลของหน้าจอ console สามารถเปลี่ยนขนาดการแสดงผลได้โดยใช้คำสั่ง SetConsoleWindowInfo() และสามารถกำหนดขนาดหน่วยความจำในการเก็บข้อมูลการแสดงผลได้โดยใช้คำสั่ง SetConsoleScreenBufferSize() โดยต้อง #include<windows.h> ก่อน หากกำหนดความกว้างและความยาวของขนาดหน้าจอเท่ากับขนาดหน่วยความจำ หน้าจอแสดงผลจะไม่มี ScrollBar ตัวอย่างโปรแกรมที่ตั้งค่าหน้าจอให้มีการแสดงผล 25 บรรทัด บรรทัดละ 80 ตัวอักษร โดยไม่มี ScrollBar ดังนี้

```
#include <windows.h>

#define screen_x 80
#define screen_y 25

HANDLE wHnd;
COORD bufferSize = { screen_x,screen_y };
SMALL_RECT windowSize = { 0,0,screen_x - 1,screen_y - 1 };

int setConsole(int x, int y)
{
      wHnd = GetStdHandle(STD_OUTPUT_HANDLE);
      SetConsoleWindowInfo(wHnd, TRUE, &windowSize);
      SetConsoleScreenBufferSize(wHnd, bufferSize);
      return 0;
}

int main()
{
      setConsole(screen_x, screen_y);
      return 0;
}
```

Example 1

เมื่อมีการรันโปรแกรมตัวอย่างจะมีการแสดงผลหน้าจอ console ขนาด 25 บรรทัด บรรทัดละ 80 ตัวอักษร และไม่มี ScrollBar

ตอนที่ 2 การแสดงผลแบบ double buffer

double buffer เป็นวิธีจัดการการแสดงผลเพื่อลดการกะพริบในช่วงของการเปลี่ยนแปลงตำแหน่งต่างๆ โดยในการแสดงผลในการทดลอง text mode game ครั้งก่อนหน้าที่มีการแสดงผลยาน และดาว บนหน้าจอโดยตรง ซึ่งจะพบปัญหาเมื่อมีการแสดงผลดาวจำนวนมาก หรือมีการเคลื่อนที่ยานอย่างรวดเร็ว (ตั้งค่า Sleep น้อยๆ) เมื่อโปรแกรมทำงานจะมีการเปลี่ยนตำแหน่งทีละจุดจนครบทั้งหน้าจอ จากการค่อยๆ เปลี่ยนตำแหน่งการแสดงผลทีละส่วนผู้ใช้งานจะรู้สึกว่ามีการเคลื่อนที่ไม่ต่อเนื่องหรือหน้าจอกะพริบ วิธีการแก้ไขปัญหาคือจะกำหนด buffer ที่มีขนาดเท่ากับหน้าจอแสดงผล แล้วทำการเตรียมข้อมูลสำหรับการแสดงผลที่ buffer ที่สร้างขึ้นก่อน หลังจากนั้นจึงทำการ copy buffer นี้ทั้งหมด ไปยังหน่วยความจำของหน้าจอพร้อมกันทั้งหมด วิธีการนี้จะมีการทำงานที่เร็วกว่าการแสดงผลข้อมูลทีละจุด ผู้ใช้งานโปรแกรมจะเห็นการเปลี่ยนแปลงพร้อมกันทั้งหน้าจอ ไม่เห็นการกะพริบ

ในการเขียนโปรแกรมให้มีการทำงานแบบ double buffer ทำได้โดยการสร้าง buffer สำหรับเตรียมการแสดงผล โดยกำหนด Array ของ struct CHAR_INFO (structure ที่มีการเก็บข้อมูล ASCII code และ attribute สี) ให้มีขนาดเท่ากับขนาดของหน้าจอ

```
CHAR_INFO consoleBuffer[screen_x * screen_y];
```

Example 2

การเตรียมข้อมูลในการแสดงผลจะส่งข้อมูลจะมีการแก้ไขข้อมูลตัวอักษรแต่ละตัวใน consoleBuffer[] ใน attribute .Char.AsciiChar และแก้ไขค่าสีและพื้นหลังใน attribute .Attributes ซึ่งการเตรียมข้อมูลสำหรับจอแสดงผลขนาด [screen_y * screen x] ให้มีการแสดงผลตัว 'A' สีขาวทั้งหมดดังนี้

```
void fill_data_to_buffer()
{
   for (int y = 0; y < screen_y; ++y) {
      for (int x = 0; x < screen_x; ++x) {
         consoleBuffer[x + screen_x * y].Char.AsciiChar = 'A';
         consoleBuffer[x + screen_x * y].Attributes = 7;
      }
   }
}
```

Example 3

หลังจากที่เตรียมข้อมูลสำหรับแสดงผลแล้ว จึงทำการถ่ายโอนข้อมูลที่ได้เตรียมไว้ส่งไปยังหน้าจอแสดงผล โดยใช้คำสั่ง WriteConsoleOutputA() ซึ่งมีการใช้คำสั่งดังนี้ (สามารถใช้คำสั่งนี้ได้โดยตรงใน main() หรือสร้างเป็นฟังก์ชั่นเพื่อเรียกใช้งานได้)

```
void fill_buffer_to_console()
{
   WriteConsoleOutputA(wHnd, consoleBuffer, bufferSize, characterPos,
&windowSize);
}
```

Example 4

โดยค่า parameter ต่างๆ ของฟังก์ชันนี้ จะเป็นตัวแปรที่กำหนดค่าไว้แล้วเกือบทั้งหมด ยกเว้นค่า characterPos ซึ่งเป็นค่าที่ระบุตำแหน่งของจอภาพมุมบนซ้ายที่จะนำข้อมูลไปแสดงผล ในการนำข้อมูลที่กำหนดไว้ใน buffer ไปแสดงผลแบบเต็มทั้งจอภาพ จึงทำการกำหนด characterPos ให้เป็นค่ามุมบนซ้ายของจอภาพคือค่า {0,0}

```
COORD characterPos = { 0,0 };
```

Example 5

ในกรณีที่ต้องการเปลี่ยนแปลงภาพที่แสดงในหน้าจอเป็นบางส่วน ก็สามารถทำได้โดยกำหนด buffer ให้มีขนาดเล็กลง แล้วกำหนดค่า characterPos ไปชี้ที่ตำแหน่งส่วนอื่นๆ ของจอภาพได้ เมื่อรวมการทำงานทั้งหมดจะได้โค้ดในการตั้งค่าจอภาพเป็นขนาด 80x25 ตัวอักษรและแสดงผลตัวอักษร 'A' สีขาวทั้งหมดดังนี้

```
#include <windows.h>
#include <time.h>

#define screen_x 80
#define screen_y 25

HANDLE wHnd;
```

```
CHAR_INFO consoleBuffer[screen_x * screen_y];
COORD bufferSize = { screen_x,screen_y };
COORD characterPos = { 0,0 };
SMALL_RECT windowSize = { 0,0,screen_x-1,screen_y-1 };

int setConsole(int x, int y)
{
   wHnd = GetStdHandle(STD_OUTPUT_HANDLE);
   SetConsoleWindowInfo(wHnd, TRUE, &windowSize);
   SetConsoleScreenBufferSize(wHnd, bufferSize);
   return 0;
}

void fill_data_to_buffer()
{
   for (int y = 0; y < screen_y; ++y) {
      for (int x = 0; x < screen_x; ++x) {
         consoleBuffer[x + screen_x * y].Char.AsciiChar = 'A';
         consoleBuffer[x + screen_x * y].Attributes = 7;
      }
   }
}

void fill_buffer_to_console()
{
     WriteConsoleOutputA(wHnd, consoleBuffer, bufferSize, characterPos,
&windowSize);
}

int main()
{
     setConsole(screen_x, screen_y);
     fill_data_to_buffer();
     fill_buffer_to_console();
     Sleep(5000);
     return 0;
}
```

Example 6

Assignment 1 : ให้นักศึกษาแก้ไขโปรแกรมใน Example 6 ให้มีการสุ่มค่าตัวอักษรเป็น A-Z และสุ่มค่าสีเป็น 0-255 แล้วมีการแสดงผลบนจอภาพทั้งหมด 10,000 รอบ

Assignment 2 : จากโปรแกรมตัวอย่าง ที่เมื่อรันโปรแกรมแล้ว จะมีการสุ่มเกิดเครื่องหมาย * จำนวน 80 ดวงกระจายอยู่เต็มจอภาพ แล้วดาวแต่ละดวงจะเคลื่อนที่ลงมาด้านล่างของ console โดยมีการเคลื่อนทั้งหมด 1000 ครั้งจึงจบการทำงาน ให้นักศึกษาเติมโปรแกรมใน 3 ฟังก์ชันต่อไปนี้ให้สมบูรณ์ โดย

- `clear_buffer()` : เป็นฟังก์ชันในการเปลี่ยนค่าใน consoleBuffer ทั้งหมดให้แสดงผลเป็นช่องว่างและมีสีเป็นสีขาวไม่มีพื้นหลัง

- `init_star()`: เป็นฟังก์ชั่นที่ทำการสุ่มพิกัดค่าเริ่มต้น {x,y} ของตัวแปร star[] ให้กระจายอยู่เต็มพื้นที่จอภาพขนาด 80x25 ตัวอักษร
- `fill_star_to_buffer()`: เป็นฟังก์ชั่นที่เติม '*' สีขาวพื้นหลังสีดำ ลงใน consoleBuffer ที่ทุกๆ ตำแหน่งใน star[]

```
#include <stdio.h>
#include <windows.h>
#include <time.h>
#define scount 80
#define screen_x 80
#define screen_y 25

HANDLE wHnd;
CHAR_INFO consoleBuffer[screen_x * screen_y];
COORD bufferSize = { screen_x,screen_y };
COORD characterPos = { 0,0 };
SMALL_RECT windowSize = { 0,0,screen_x-1,screen_y-1 };
COORD star[scount];

int setConsole(int x, int y)
{
      wHnd = GetStdHandle(STD_OUTPUT_HANDLE);
      SetConsoleWindowInfo(wHnd, TRUE, &windowSize);
      SetConsoleScreenBufferSize(wHnd, bufferSize);
      return 0;
}

void clear_buffer()
{
       ...
}

void fill_buffer_to_console()
{
      WriteConsoleOutputA(wHnd, consoleBuffer, bufferSize, characterPos,
&windowSize);
}

void init_star()
{
       ...
}

void star_fall()
{
      int i;
      for (i = 0; i < scount; i++) {
            if (star[i].Y >= screen_y-1) {
                  star[i] = { (rand() % screen_x),1 };
            }
            else {
                  star[i] = { star[i].X,star[i].Y+1 };
            }
      }
```

```
}


void fill_star_to_buffer()
{
      ...
}

int main()
{
      int i;
      srand(time(NULL));
      setConsole(screen_x, screen_y);
      init_star();

      i = 0;
      while (i < 1000)
      {
            star_fall();
            clear_buffer();
            fill_star_to_buffer();
            fill_buffer_to_console();
            Sleep(200);
            i++;
      }
      return 0;
}
```

Example 7

ตอนที่ 3 การใช้ mouse และ keyboard จาก Console input events

จากโค้ดเริ่มต้นใน Example 1 สามารถเขียนโปรแกรมเพิ่มเติมเพื่อรับ input จาก mouse และ keyboard ได้ตามขั้นตอนดังนี้

1. ทำการตั้งค่า console mode โดยใช้ฟังก์ชัน SetConsoleMode() ซึ่งจะตั้งค่าโหมดการทำงานให้โปรแกรมสามารถอ่านค่า windows input และ mouse input ได้

```
HANDLE rHnd;
DWORD fdwMode;
…

int setMode()
{
      rHnd = GetStdHandle(STD_INPUT_HANDLE);
      fdwMode = ENABLE_EXTENDED_FLAGS | ENABLE_WINDOW_INPUT |
ENABLE_MOUSE_INPUT;
      SetConsoleMode(rHnd, fdwMode);
      return 0;
}
…

int main()
```

```
{
      setConsole(screen_x, screen_y);
      setMode();
      return 0;
}
```

Example 8

จากตัวอย่างคำสั่ง `SetConsoleMode(rHnd, fdwMode);` มีการใช้ parameter 2 ตัวคือ rHnd ซึ่งเป็นตัวแปรที่เก็บค่า handle ของจอภาพจากคำสั่ง `GetStdHandle()` และ `fdwMode` ที่เป็นค่าตัวแปรสำหรับตั้งค่าการอ่าน windows input และ mouse input

```
DWORD numEvents = 0;
DWORD numEventsRead = 0;
…

GetNumberOfConsoleInputEvents(rHnd, &numEvents);
if (numEvents != 0) {
   INPUT_RECORD* eventBuffer = new INPUT_RECORD[numEvents];
   ReadConsoleInput(rHnd, eventBuffer, numEvents, &numEventsRead);
   for (DWORD i = 0; i < numEventsRead; ++i) {
    // … eventBuffer check and process …
   }
}
```

Example 9

2. อ่านค่าจำนวน input event ที่เกิดขึ้น โดยใช้คำสั่ง GetNumberOfConsoleInputEvents (rHnd, &numEvents); ซึ่งจะใช้ parameter 2 ตัวคือ rHnd (ตัวแปรเก็บ console handler) และเมื่อฟังก์ชันทำงานเสร็จแล้วจะส่งค่าเก็บไว้ในตัวแปร numEvents
3. ทำการตรวจสอบค่า numEvents ว่ามีค่ามากกว่า 0 ในกรณีที่ค่า numEvents มีค่ามากกว่า 0 ให้อ่าน input ทั้งหมด มาเก็บไว้ใน buffer สำหรับเก็บ event (eventBuffer) โดยใช้คำสั่ง `ReadConsoleInput()` ซึ่ง eventBuffer จะเป็น array ที่ถูกสร้างขึ้นโดยมีขนาดเท่ากับจำนวน numEvents
4. ในส่วนของโค้ด eventBuffer check and process ภายในลูป จะเป็นส่วนของการตรวจสอบค่า eventBuffer แต่ละค่าว่าเป็น event ของอะไรซึ่งจะมีตัวแปรที่สำคัญดังนี้

- `eventBuffer[i].EventType` มีค่าเป็น `KEY_EVENT` หรือ `MOUSE_EVENT`
- `eventBuffer[i].Event.KeyEvent.bKeyDown` เป็นค่าสำหรับตรวจสอบว่ามีการกดคีย์บอร์ดลงหรือไม่
- `eventBuffer[i].Event.KeyEvent.wVirtualKeyCode` เป็นค่า key code ของ key ที่กด เช่น `VK_ESCAPE` เป็นค่าของปุ่ม ESC
- `eventBuffer[i].Event.KeyEvent.uChar.AsciiChar` เป็นค่า ASCII Code ของ key ที่กด
- `eventBuffer[i].Event.MouseEvent.dwMousePosition.X` เป็นค่าพิกัดในแกน x ของ mouse
- `eventBuffer[i].Event.MouseEvent.dwMousePosition.Y` เป็นค่าพิกัดในแกน y ของ mouse

- `eventBuffer[i].Event.MouseEvent.dwButtonState` เป็นสถานะของการกดปุ่มของ mouse ซึ่งจะเป็นค่า `FROM_LEFT_1ST_BUTTON_PRESSED` สำหรับการคลิ๊กปุ่มซ้าย `RIGHTMOST_BUTTON_PRESSED` สำหรับคลิ๊กปุ่มขวา
- `eventBuffer[i].Event.MouseEvent.dwEventFlags` เป็นสถานะของ mouse ซึ่งเช่นมีค่าเป็น `MOUSE_MOVED` เมื่อมีการเคลื่อน mouse

จากรายละเอียดทั้งหมดจะสามารถสร้างโปรแกรมที่ตรวจสอบการกดปุ่ม keyboard , การคลิ๊กเมาส์ และการเคลื่อนที่ของเมาส์ โดยจะแสดงผลคีย์ที่กด พิกัดของเมาส์เมื่อเมาส์เคลื่อนที่ รวมถึงการคลิ๊กเมาส์ปุ่มซ้ายและปุ่มขวา ดังตัวอย่าง

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

#define screen_x 80
#define screen_y 25

HANDLE rHnd;
HANDLE wHnd;
DWORD fdwMode;
COORD bufferSize = { screen_x,screen_y };
SMALL_RECT windowSize = { 0,0,screen_x-1,screen_y-1 };

… // function setConsole() and setMode()

int main()
{
  bool play = true;
  DWORD numEvents = 0;
  DWORD numEventsRead = 0;
  setConsole(screen_x, screen_y);
  setMode();
  while (play)
  {
    GetNumberOfConsoleInputEvents(rHnd, &numEvents);
    if (numEvents != 0) {
      INPUT_RECORD* eventBuffer = new INPUT_RECORD[numEvents];
      ReadConsoleInput(rHnd, eventBuffer, numEvents, &numEventsRead);
      for (DWORD i = 0; i < numEventsRead; ++i) {
        if (eventBuffer[i].EventType == KEY_EVENT &&
            eventBuffer[i].Event.KeyEvent.bKeyDown == true ) {
          if (eventBuffer[i].Event.KeyEvent.wVirtualKeyCode == VK_ESCAPE) {
            play = false;
          }
          printf("press : %c\n", eventBuffer[i].Event.KeyEvent.uChar.AsciiChar);
        }
        else if (eventBuffer[i].EventType == MOUSE_EVENT) {
          int posx = eventBuffer[i].Event.MouseEvent.dwMousePosition.X;
          int posy = eventBuffer[i].Event.MouseEvent.dwMousePosition.Y;
          if (eventBuffer[i].Event.MouseEvent.dwButtonState &
              FROM_LEFT_1ST_BUTTON_PRESSED) {
            printf("left click\n");
          }
          else if (eventBuffer[i].Event.MouseEvent.dwButtonState &
                   RIGHTMOST_BUTTON_PRESSED) {
                printf("right click\n");
          }
```

```
            else if (eventBuffer[i].Event.MouseEvent.dwEventFlags & MOUSE_MOVED) {
                  printf("mouse position : (%d,%d)\n",posx, posy);
            }
          }
        }
        delete[] eventBuffer;
      }
    Sleep(100);
    }
    return 0;
}
```

Example 10

Assignment 3 : ให้นักศึกษาเพิ่มเติมโปรแกรมให้มีการแสดงผลยานอยู่ตรงตำแหน่ง mouse และมีการเปลี่ยนสีแบบสุ่ม เมื่อมีการกดปุ่ม ‘c’ หรือกดเมาส์ปุ่มซ้าย

Assignment 4 : ให้นักศึกษาเพิ่มโปรแกรมจาก Assignment 3 ให้มีการสุ่มตำแหน่งการแสดงผล “*” บนจอภาพ ถ้ายานเคลื่อนที่ไปชน “*” จะมีการสุ่มเพื่อเปลี่ยนตำแหน่ง “*” และโปรแกรมจะหยุดการทำงานเมื่อมีการกดปุ่ม ESC หรือมีการชน “*” 10 ครั้ง

การส่งงาน : ให้นักศึกษาส่งเฉพาะ Assignment 4