Advanced Programming

Assignment #3

Submission date : 2019. 6. 6

Affiliation : Department of Computer Software

Student ID : 2017203053

Student Name : 김 형 석

1. Summary of my results

Random	Beehive	Glider	Oscillator	Seeds	Replicator
O	O	O	O	O	O

2. Basic / Advanced Requirement

(1) Basic

① class BinaryBitmap

private:

int* width; // 바둑판의 사이즈가 달라질 때마다 새롭게 pixel 개수를 받기 위함

int* height;

public:

bool** map; // 각 pixel의 bool 값을 동적으로 저장하기 위한 변수

- BinaryBitmap(int w, int h); // Constructor
  : main 코드에서 입력받은 사이즈에 해당하는 맵을 동적으로 할당

- ~BinaryBitmap(); // Destructor
  : 객체가 사용되고 난 후 동적 할당된 변수들의 메모리를 해제 시켜줌

- BinaryBitmap(const BinaryBitmap& bb); // Copy Constructor
  : 제가 작성한 코드에서는 복사 생성자가 사용 되지 않았습니다. 하지만 복사에 해당하는 함수 정의는 작성되있습니다.
- BinaryBitmap& operator=(const BinaryBitmap& bb);
  : Random case를 생성할 때 원본에 저장된 pixel 값들을 똑같이 불러오는 역할
- void clear();
  : 사이즈를 바꾸거나 새로운 이미지를 불러올 때 화면을 다 지우고 다시 값 저장
- bool get(int x, int y) const;
  : 각 pixel에 저장된 bool 값을 반환 해줌
- void set(int x, int y, bool v);
  : 각 pixel에 bool 값을 저장함
- int getWidth() const;
  : private에 선언되있는 width 값을 저장
- int getHeight() const;
  : private에 선언되있는 height 값을 저장

② class CellularAutomata

- enum class InitType { CLEAN, RANDOM, BEEHIVE, GLIDER, OSCILLATOR };
  : 각 case에 해당하는 버튼에 해당하는 값

- CellularAutomata(int w, int h); // Constructor
  : CellularAutomata 내에 선언된 BinaryBitmap 객체에서 main에서 전달된 값에 해당하는 사이즈의 맵 동적으로 생성

- virtual ~CellularAutomata(); // Virtual Destructor
  : CellulartAutomata 클래스의 상속을 받는 클래스에서도 실행되도록 virtual로 선언됨
    CellularAutomata 클래스 내에 동적으로 할당된 BinaryBitmap 객체 해제
- void initialize(InitType t);
  : 각 Case에 해당하는 이미지 불러옴
- void clear();
  : BinaryBitmap의 clear 함수 불러옴
- void resize(int w, int h);
  : 현재 생성되있는 맵을 파기하고 새롭운 사이즈로 재할당
- bool copy(const CellularAutomata* automata);
  : Random case 일 때 원본의 맵을 복사해서 CellularAutomata에서 생성된 맵에 저장
- virtual void update();
  : CellularAutomata로부터 상속받은 class에서 정의된 update()함수 불러옴
- void set(int x, int y, bool v);
  : BinaryBitmap 클래스의 set 함수 불러옴
- bool get(int x, int y) const;
  : BinaryBitmap 클래스의 get 함수 불러옴
- int getWidth() const;
  : main 코드에서 전달된 너비와 값을 CellularAutomata에서 선언된
    BinaryBitmap 객체에 전달
- int getHeight() const;
  : main 코드에서 전달된 높이 값을 CellularAutomata에서 선언된
    BinaryBitmap 객체에 전달

③ class LifeAutomata
- LifeAutomata(int w, int h); // Constructor
  : CellularAutomata 생성자와 동일
- void update() override;
  : 각 case에 해당하는 이미지를 conway rule에 맞게 갱신

(2) Advanced
① class SeedAutomata
  : LifeAutomata 클래스와 update에서 실행되는 rule만 다름
② class ReplicatorAutomata
  : LifeAutomata 클래스와 update에서 실행되는 rule만 다름

3. 고찰

이번 과제를 시작할 때 제공된 Class의 public interface를 먼저 쭉 나열한 후 제공된 main driver 소스파일을 보면서 각 클래스의 멤버 함수들이 어떤 식으로 호출이 되는지 비교를 했습니다. 처음에는 BinaryBitmap 클래스에서 CellularAutomata 객체를 생성해야 하는지 아니면 CellularAutomata 클래스에서 BinaryBitmap 객체를 생성해야하는지 조차 헷갈렸는데 main 코드를 보면서 조금씩 진전이 생겼습니다. 제가 잘 모르는 문법들이 많아서 거의 멤버 함수 하나를 정의 할 때마다 그에 해당하는 문법을 공부한 후 정의를 해서 약간의 시간은 더 걸렸을지 몰라도 과제를 하면서 문법 공부를 하게 되어 큰 도움이 되었습니다. 기초 지식이 조금 더 좋았더라면 주어진 멤버함수를 모두 잘 활용해서 코드를 작성했을텐데 제가 아는 선에서 맞춰 작성하다보니 멤버함수들을 제대로 활용하지 못한거같아 아쉬웠습니다.

이번 과제에서 어려움을 겪었던 부분은 각 pixel의 값들을 bool 값으로 저장하는 것이였던거 같습니다. 제공된 public interface에서 각 pixel 값을 저장하는 변수가 없어서 일단 private 영역에 2차원 배열 선언해서 작성을 했는데 CellularAutomata 클래스에서 BinaryBitmap 클래스에서 저장된 값을 호출할 때 제대로 호출을 할 수 없어서 public 영역에 선언을 하게 됐습니다. 어려움을 겪었던 또다른 부분은 동적할당 부분이였습니다. 멤버함수에서 동적할당과 해제를 하는 함수가 많은데 메모리 접근 부분에서 할당된 메모리에 맞게 접근해야되는데 잘못 접근이 돼서 발생되는 컴파일 에러가 어디서 잘못 접근이 됐는지 찾아내기가 쉽지 않았습니다.

처음 과제 pdf를 봤을 때는 pdf의 페이지 수를 보고 과연 이 과제를 해낼 수 있을까 하는 걱정이 굉장히 컸는데 막상 과제를 끝내고 나니 제공된 코드에 맞춰 함수 정의만 하면 되는 그리 어렵지 않은 문법 과제라는 생각이 들었습니다. 이 과제를 하면서 스스로 뿌듯했던 단계가 있는데 첫 번째는 UI 콘솔창을 처음 에러 없이 띄웠을 때고 두 번째는 pixel이 룰에 따라 정상적으로 바뀌게 하는 것에 성공 했을 때 라고 생각됩니다.