2021-1 C++프로그래밍 실습과제 08

(1) 각 문제에 대한 분석과 및 해결 방법

8.1. 8.8절의 Monster World 프로그램을 다음과 같이 확장하라.

(0) 먼저 8.8절의 Monster World 프로그램을 구현하고 동작을 확인하라.

(1) 이 프로그램의 Monster클래스에 실습문제 7.3과 같이 "에너지"를 나타내는 속성을 추가하고, 관련 함수들을 수정하라. Monster의 생성자와 eat(), print()함수를 수정하면 된다.

(2) Monster클래스에 "에너지" 값을 반환하는 getEnergy()함수를 추가하라. 이 함수는 private 멤버인 nEnergy를 반환한다.

(3) 이제 에너지가 0이 되면 몬스터가 사라지도록 프로그램을 수정하라. 이를 위해 MonsterWorld클래스에 checkStarvation() 함수를 추가한라. 이 함수에서는 모든 몬스터의 에너지 레벨을 검사하여 0인 몬스터들을 모두 동적으로 해제한다. 특히 pMon 배열에서 삭제할 항목의 처리에 유의해야 한다. 힌트: k번째 항목을 삭제하는 경우 맨 마지막 항목을 k번째 항목에 복사하고 몬스터의 수를 1 감소하면 된다.

(4) Monster클래스에 몬스터의 수를 나타내는 정적 멤버 변수를 추가하라. 또한 생성자와 소멸자에서 이 멤버 변수의 값을 적절히 갱신하고, 현재 몬스터의 수를 출력하는 정적 멤버 함수 printCount()를 구현하라. 화면을 갱신할 때 마다 이 함수를 호출하여 전체 몬스터의 수를 출력한다. (이 정적 변수의 값은 MonsterWorld의 nMon과 동일하도록 유지되어야 한다.)

(5) 화면 출력은 다음의 예를 참고하여 구성하라.

[문제분석 및 해결방법] : 힌트를 참고하여 에너지가 0인 몬스터들의 몬스터 포인터 변수를 이용해 할당을 해제하고 스왑하는 방식으로 checkStarvation() 함수를 구현하였다. 또한 몬스터가 죽었을 때 상황을 확인할 수 있도록 getchar()를 적절히 삽입하였다. 몬스터 객체에서 nEnergy값을 가져올 수 있도록 getter함수인 getEnergy()함수를 추가하였고, Monster 클래스에 몬스터 수를 나타내는 정적 멤버 변수인 nMonster와 정적 멤버 함수 printCount의 정의를 추가하고 cpp 파일에서 초기화하여 출력 화면에서 몬스터의 현재 수가 표시될 수 있도록 구현하였다. 기존의 int map[DIM][DIM] 변수를 제거하고 Matrix 클래스를 이용해 내부적으로 동적 할당을 이용한 격자맵의 처리가 이루어지도록 코드를 수정하였다.

(2) 자신이 구현한 주요 코드

8.1. 8.8절의 Monster World 프로그램을 다음과 같이 확장하라.

<LJH_08_1.cpp>

```cpp
#include "MonsterWorld.h"
#include <time.h >
int Monster::nMonster =0;
void Monster::printCount() {
  cout <<"  전체 몬스터의 수 : "<< Monster::nMonster <<endl;
}
int main()
{
  srand((unsigned int)time(NULL));
  int w =20, h =10;
  MonsterWorld game(w, h);
  game.add(new Monster("Integral", "∫", rand() % w, rand() % h));
  game.add(new Monster("PartialD", "∂", rand() % w, rand() % h));
  game.add(new Monster("Delta", "Δ", rand() % w, rand() % h));
  game.add(new Monster("Del", "▽", rand() % w, rand() % h));
  game.add(new Monster("Plus", "+", rand() % w, rand() % h));
  game.play(500, 10);
  printf("------게임 종료--------------------₩n");
  std::cout <<std::endl <<"Press ENTER to exit..."; fflush(stdin); getchar();
  return 0;
}
```

<Monster.h>

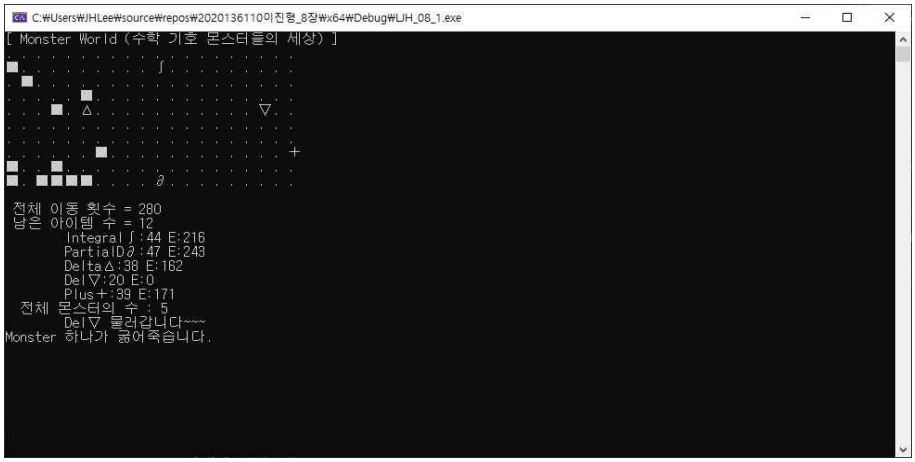
```cpp
#pragma once
#include "Canvas.h"
#define DIM 40
class Monster {
  string name, icon;
  int x, y, nItem;
  int nEnergy;
  static int nMonster;
  ~ ... ~
  int getEnergy() {
    return nEnergy;
  }
  static void printCount();
};
```

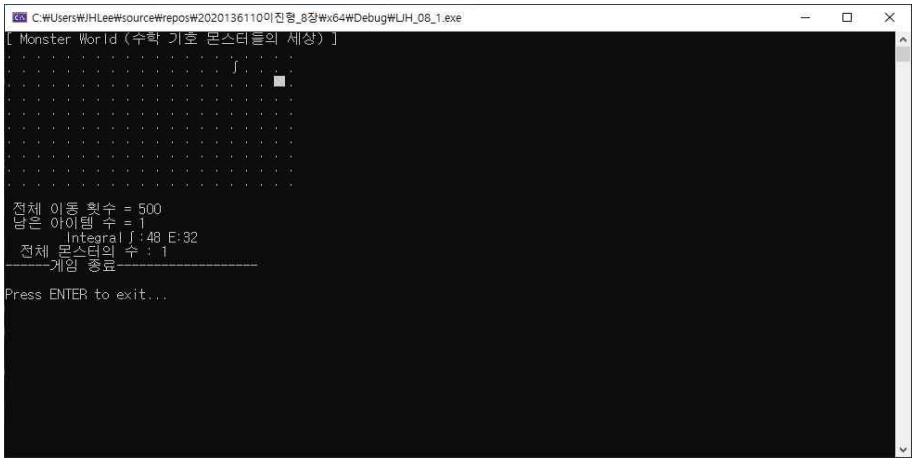
<MonsterWorld.h>

```
#pragma once
#include "Canvas.h"
#include "Monster.h"
#include "Matrix.h"
#include <windows.h >
#define DIM  40
#define MAXMONS 5
class MonsterWorld {
  Matrix world;
  int xMax, yMax, nMon, nMove;
  Monster* pMon[MAXMONS];
  ~ ... ~
  void print() {
    canvas.clear();
    for (int y =0; y < yMax; y ++)
      for (int x =0; x < xMax; x ++)
        if (Map(x, y) >0) canvas.draw(x, y, "■");
    for (int i =0; i < nMon; i ++)
      pMon[i]->draw(canvas);
    canvas.print("[ Monster World (수학 기호 몬스터들의 세상) ]");
    cerr <<" 전체 이동 횟수 = "<< nMove <<endl;
    cerr <<" 남은 아이템 수 = "<< countItems() <<endl;
    for (int i =0; i < nMon; i ++)
      pMon[i]->print();
    Monster::printCount();
  }
  void checkStarvation()
  {
    for (int i =0; i < nMon; i ++)
    {
      if (pMon[i]->getEnergy() ==0)
      {
        delete  pMon[i];
        pMon[i] = pMon[nMon -1];
        nMon--;
        cerr <<"Monster 하나가 굶어죽습니다."<<endl;
        getchar();
      }
    }
  }
public:
  MonsterWorld(int w, int h) : world(h, w), canvas(w, h), xMax(w), yMax(h) {
     ~ ... ~
  }
  ~MonsterWorld() {
    for (int i =0; i < nMon; i ++)
        delete pMon[i];
    }
  }
  ~ ... ~
  void play(int maxwalk, int wait) {
    print();
    cerr <<" 엔터를 누르세요...";
    getchar();
    for (int i =0; i < maxwalk; i ++) {
      for (int k =0; k < nMon; k ++)
        pMon[k]->move(world.Data(), xMax, yMax);
      nMove++;
      checkStarvation();
      print();
      if (isDone()) break;
      Sleep(wait);
    }
  }
};
```

(3) 다양한 입력에 대한 테스트 결과

8.1. 8.8절의 Monster World 프로그램을 다음과 같이 확장하라.





(4) 코드에 대한 설명 및 해당 문제에 대한 고찰

8.1. 8.8절의 Monster World 프로그램을 다음과 같이 확장하라.

(2) Monster 클래스에 "에너지" 값을 반환하는 getEnergy()함수를 추가하라. 이 함수는 private 멤버인 nEnergy를 반환한다.

<Monster.h>

```
#pragma once
#include "Canvas.h"
#define DIM 40
class Monster {
  string name, icon;
  int x, y, nItem;
  int nEnergy;
  ~ ... ~
  int getEnergy() {
    return nEnergy;
  }
  ~ ... ~
};
```

Monster 클래스에 에너지 값을 반환하는 getter인 int getEnergy() 함수를 추가하고 클래스 내부 private 멤버인 nEnergy를 반환하도록 구성하였다.

(3) 이제 에너지가 0이 되면 몬스터가 사라지도록 프로그램을 수정하라. 이를 위해 MonsterWorld클래스에 checkStarvation() 함수를 추가한라. 이 함수에서는 모든 몬스터의 에너지 레벨을 검사하여 0인 몬스터들을 모두 동적으로 해제한다.

<MonsterWorld.h>

```
#pragma once
#include "Canvas.h"
#include "Monster.h"
#include "Matrix.h"
#include <windows.h >
#define DIM   40
#define MAXMONS 5
class MonsterWorld {
  Matrix world;
  int xMax, yMax, nMon, nMove;
  Monster* pMon[MAXMONS];
  ~ ... ~
  void checkStarvation()
  {
    for (int i =0; i < nMon; i ++)
    {
      if (pMon[i]->getEnergy() ==0)
      {
        delete  pMon[i];
        pMon[i] = pMon[nMon -1];
        nMon--;
        cerr <<"Monster 하나가 굶어죽습니다."<<endl;
        getchar();
      }
    }
  }
public:
  ~ ... ~
  void play(int maxwalk, int wait) {
    print();
    cerr <<" 엔터를 누르세요...";
    getchar();
    for (int i =0; i < maxwalk; i ++) {
      for (int k =0; k < nMon; k ++)
        pMon[k]->move(world.Data(), xMax, yMax);
      nMove++;
      checkStarvation();
      print();
      if (isDone()) break;
      Sleep(wait);
    }
  }
};
```

MonsterWorld 클래스에 굶어죽은 몬스터를 할당 해제하고 앞으로 나타나지 않게 포인터 배열에서 제거하고 그 결과를 게임에서 확인할 수 있도록 getchar() 함수를 이용하여 사용자의 키 입력을 기다리게끔 checkStarvation() 함수를 구현하였다.

(4) Monster클래스에 몬스터의 수를 나타내는 정적 멤버 변수를 추가하라. 또한 생성자와 소멸자에서 이 멤버 변수의 값을 적절히 갱신하고, 현재 몬스터의 수를 출력하는 정적 멤버 함수 printCount()를 구현하라. 화면을 갱신할 때 마다 이 함수를 호출하여 전체 몬스터의 수를 출력한다.

<Monster.h>
```
#pragma once
#include "Canvas.h"
#define DIM 40
class Monster {
  string name, icon;
  int x, y, nItem;
  int nEnergy;
  static int nMonster;
  ~ ... ~
  static void printCount();
};
```

<LJH_08_1.cpp>
```
#include "MonsterWorld.h"
#include <time.h>
int Monster::nMonster =0;
void Monster::printCount() {
  cout <<"  전체 몬스터의 수 : "<< Monster::nMonster <<endl;
}
~ ... ~
```

Monster 클래스에 정적 멤버 변수와 함수인 nMonster, void printCount()를 추가하여 현재 남은 전체 몬스터의 수가 표시될 수 있도록 하였다.

(5) 이번 과제에 대한 느낀점

이번 과제를 해결하기 위해 교재의 코드를 분석하고 조사하면서 모르고 있었던 세부적인 내용을 많이 알 수 있어서 좋았다.