프로젝트 최종 보고서

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. 프로그램 개요   * 포켓몬 배틀 프로그램 * 포켓몬 생산, 배틀 턴, 공격, 휴식, 스킬 사용 등 * 포켓몬 배틀 프로그램에서 수행할 수 있는 기능들을 포함   .  2. 전체 논리 구조도  3. 모듈 별 기능   * + 1. App        1. 아군 포켓몬 – 사용자의 입력을 받아 포켓몬을 생산하여 AllyList 벡터에 저장함        2. 적군 포켓몬 – 사용자의 입력을 받아 포켓몬을 생산하여 Enemy 벡터에 저장함        3. 턴 – 순서대로 공격과 방어를 진행하기 위해 턴을 설정함.     2. TurnBehavior        1. 공격 – 이번 턴에 일반 공격을 수행하며 데미지는 미리 설정된 수식을 이용함. 이때 예외 케이스와 상성을 고려함.        2. 휴식 – 이번 턴에 공격하지 않고 체력을 일정량만큼 회복함. 이때 현재 체력이 체력의 최대값을 넘지 않도록 함.        3. 스킬 – 포켓몬마다 주어진 스킬을 배틀 당 최대 2번까지 사용할 수 있음.     3. PokemonFactory        1. 포켓몬 생성 – 생성하고 싶은 포켓몬 ID를 입력 받아 포켓몬 객체를 생성하고 주소값을 반환.     4. Pokemon        1. 포켓몬 객체로 각자 주어진 ID, 이름, 체력, 공격력, 방어력, 타입, 스킬 등을 멤버 변수로 가짐.        - 프로세스 처리도 & 시나리오 흐름도     5. 모듈의 함수     |  | | --- | | **App.h** | | #pragma once  #include <vector>  #include "AttackBehavior.h"  #include "RestBehavior.h"  #include "SkillBehavior.h"  #include "PokemonFactory.h"  class App  {  public:  App(); // 생성자  ~App(); // 소멸자  void run(); // 프로그램 실행  private:  void startBattle();// 아군 vs 적군 싸움  void selectAllyPokemon(int type); // 아군 포켓몬 선택  void selectEnemyPokemon();// 적군 포켓몬 랜덤 선택  void eraseAlly(); // 체력이 0인 아군 포켓몬을 리스트에서 삭제  void eraseEnemy(); // 체력이 0인 적군 포켓몬을 리스트에서 삭제  void printStart();// 시작화면  void printAllyPokemon(); // 아군 포켓몬 리스트 출력  void selectTurnBehavior(); // 현재 턴에 할 행동 선택  void executeTurn(); // 턴 수행 및 다음 턴으로 진행  private:  vector<Pokemon\*> allyList; // 아군 포켓몬 저장  vector<Pokemon\*> enemyList; // 적군 포켓몬 저장  PokemonFactory pf; // 포켓몬 팩토리 객체  TurnBehavior \*tb;  Pokemon\* curAlly; // 현재 배틀 중인 아군 포켓몬 객체  Pokemon\* curEnemy; // 현재 배틀 중인 적군 포켓몬 객체  bool isMyTurn = true; // 자신의 턴을 확인하는 플래그  }; |  |  | | --- | | **Pokemon.h** | | #pragma once  #include <string>  #include <iostream>  using namespace std;  class Pokemon  {  public:  Pokemon();  ~Pokemon();  virtual int skill(Pokemon\* ally, Pokemon\* enemy) = 0; // 포켓몬의 스킬을 자식 객체에서 정의  void printStatus();  public:  int type1; // 지상, 비행 속성 값  string name; // 이름  int maximum\_health\_point; // 최대 체력  int health\_point; // 현재 체력  int attack\_point; // 공격력  int defense\_point; // 방어력  int skill\_cnt; // 스킬사용횟수  }; |  |  | | --- | | **PokemonFactory.h** | | #pragma once  #include "Pokemon.h"  #include "Charizard.h"  #include "Bulbasaur.h"  #include "Blastoise.h"  #include "Pidgey.h"  #include "Metapod.h"  #include "Magikarp.h"  class PokemonFactory  {  public:  PokemonFactory();  ~PokemonFactory();  Pokemon\* create\_pokemon(int pokemon\_id); // 포켓몬 생산  }; |  |  | | --- | | **TurnBehavior.h** | | #pragma once  #include "Pokemon.h"  class TurnBehavior  {  public:  TurnBehavior();  ~TurnBehavior();  virtual int executeTurn(Pokemon\* subject, Pokemon\* target) = 0; // 이번 턴에 할 행동을 자식 클래스에서 결정  }; |   6. 설계 원칙  설계 요구 조건에 따라 충실히 설계한다.   1. 프로그램을 구현하기 위한 클래스는 App, TurnBehavior, PokemonFactory, Fokemon이다. 2. Fokemon 클래스는 포켓몬스터의 공통적인 특성을 갖는 부모 클래스이며, 이 클래스를 상속받아 다양한 종류의 포켓몬스터들을 위한 클래스를 만든다. 3. TurnBehavior 클래스는 포켓몬 게임을 위한 포켓몬들의 행동(공격, 스킬사용, 휴식)을 위한 것이며, 이 클래스를 상속받아 각각의 행동을 위한 자식 클래스를 만든다. 4. PokemonFactory 클래스는 다양한 종류의 포켓몬스터들의 객체를 생성하는 클래스이다. 5. App 클래스는 게임 프로그램 전체를 관리하는 클래스이며, 적군 포켓몬스터들을 담는 컨테이너와, 아군 포켓몬스터들을 담는 컨테이너를 맴버변수로 갖는다. 6. 포켓몬스터들의 행동에 따른 클래스 구조 설계를 위하여 전략패턴을 위한 TurnBehavior의 개체를 맴버변수로 가지고 있으며, 포켓 몬스터들의 생성을 위한 구조 설계를 위하여 팩토리 패턴을 위한 PokemonFactory 객체를 맴버변수로 갖는다.   7. 개발도구  Microsoft Visual Studio 2013 C++  8. 사용 매뉴얼     * 포켓몬 선택     사용자가 사용할 포켓몬과 적이 사용할 포켓몬을 차례대로 3마리씩 선택함   * 포켓몬 출력     사용자와 적이 각각 3마리씩 포켓몬 선택을 끝냈다면 선택된 현황을 화면에 출력    양 쪽 모두 선택이 끝났다면 선발된 포켓몬을 출력하고 배틀이 시작됨을 알림   * 나의 턴     현재 배틀 중인 포켓몬 현황을 출력함과 동시에 이번 턴에 진행할 행동을 입력받음   * 공격       AttackBehavior 에 있는 주어진 데미지 계산 공식에 따라 상대방에게 데미지를 입힘  이때 상성 상 우위인 경우 “효과는 대단했다”를, 열세인 경우“효과가 별로인 듯 하다”를 출력   * 휴식             RestBehavior 에 있는 회복 공식에 따라 체력을 회복  현재 체력이 최대 체력을 넘지 못함   * 스킬         각 포켓몬 객체에게 설정된 스킬을 사용함  효과를 적용 시킨 후 잔여 스킬 횟수를 감소시킴  스킬 횟수가 0일 때부터는 스킬을 사용할 수 없음   * 행동 불능 상태       Queue의 형태로 삭제함   * 게임 종료     PokemonList의 크기가 0이 되면 승리 또는 패배를 출력 |