하위 설계서

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | 1. 목적 |  * 고급객체지향 프로그래밍의 실습 과제이다. 클래스, 상속, 자료구조 등을 사용한 프로그래밍을 통하여 실력을 향상시킨다. * 실전 프로그래밍에서의 예외 처리, 다양한 선택변수 등을 접하고 대비하는 능력을 향상시킨다.  |  | | --- | | 1. 사용자 Targeting |  * 포켓몬 배틀 프로그램을 이용하고자 하는 고객 및 관리자를 대상으로 하며, 직관적인 인터페이스로 누구나 쉽게 포켓몬 배틀 프로그램을 이용할 수 있어야 한다.  |  | | --- | | 1. 설계 원칙 |  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.  |  | | --- | | 1. 개발 도구 |   Microsoft Visual Studio C++ 2013   |  | | --- | | 1. 적용 범위 |  * 장소적 범위 : 포켓몬 배틀 프로그램이 사용될 수 있는 장소(게임이 제공되는 장소 등) * 프로그램적 범위 : 스마트 폰 어플리케이션 등  |  | | --- | | 1. 시나리오 흐름도 |   시나리오 흐름도     |  | | --- | | 1. 인터페이스 정의 |  * 문장은 사용자가 해당 기능에 대한 사용법을 명확히 알 수 있도록 명시한다. * 글자가 지저분하지 않도록 띄어쓰기와 줄 바꿈을 해준다. * 입력은 키보드로만 받는다. * 한국어와 영어 외의 언어는 사용하지 않는다.  |  | | --- | | 1. 헤더와 cpp, 클래스정의 |  * 클래스 선언은 헤더파일을 만들고 메인 cpp에 불러온다. * 헤더에서 선언한 클래스는 헤더와 같은 이름의 cpp에 구현한다.  |  |  | | --- | --- | | **헤더와 cpp** | | | 기능 | 이름 | | 클래스 선언 | App.h  AttackBehavior.h  Blastoise.h  Bulbasaur.h  Charizard.h  Magikarp.h  Metapod.h  Pidgey.h  Pokemon.h  PokemonFactory.h  RestBehavior.h  SkillBehavior.h  TurnBehavior.h | | 클래스 구현 | App.cpp  AttackBehavior.cpp  Blastoise.cpp  Bulbasaur.cpp  Charizard.cpp  Magikarp.cpp  Metapod.cpp  Pidgey.cpp  Pokemon.cpp  PokemonFactory.cpp  RestBehavior.cpp  SkillBehavior.cpp  TurnBehavior.cpp | | 메인 구현 | Main.cpp |  |  |  | | --- | --- | | **클래스** | | | 클래스 이름 | 기능 | | **App.h** | 전체 기능 총괄 | | AttackBehavior.h | 일반 공격을 수행 | | Blastoise.h | 거북왕의 능력치 | | Bulbasaur.h | 이상해씨의 능력치 | | Charizard.h | 리자몽의 능력치 | | Magikarp.h | 잉어킹의 능력치 | | Metapod.h | 단데기의 능력치 | | Pidgey.h | 피죤의 능력치 | | **Pokemon.h** | 포켓몬 | | **PokemonFactory.h** | 아군 및 적군의 포켓몬을 생산 | | RestBehavior.h | 이번 턴에 공격하지 않고 체력을 회복 | | SkillBehavior.h | 현재 포켓몬의 스킬을 사용 | | **TurnBehavior.h** | 각 턴에 수행할 수 있는 명령 |   ▣ 클래스 별 요소와 기능 ▣   |  | | --- | | **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  }; |  |  | | --- | | 1. Main의 변수와 cpp정의 |   App : App 클래스 선언   |  | | --- | | 1. 프로젝트 추진 일정 |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  | **추진일정** | | | | | | | | | | | | **1주** | | **2주** | | **3주** | | **4주** | | **5주** | | | | **요구사항 분석** |  |  |  |  |  |  |  |  |  |  |  | | **헤더,  cpp작성** |  |  |  |  |  |  |  |  |  |  |  | | **main작성** |  |  |  |  |  |  |  |  |  |  |  | | **1차 완성** |  |  |  |  |  |  |  |  |  |  |  | | **옵션 추가** |  |  |  |  |  |  |  |  |  |  |  | | **버그 수정** |  |  |  |  |  |  |  |  |  |  |  | | **검토 및  제출** |  |  |  |  |  |  |  |  |  |  |  | |