Финален проект по ооп практикум

Изготвил: Петра Росенова Рашева

Фн 45537 , 2 група, Информатика

***WORLD OF WARCRAFT***

1. Описание и идея на проекта

Да се реализира прототип на батъл играта World of Warcraft. За целта са реализирани герои, противници и поле за битка.След като потребителят избере герой, с когото иска да се дуелира, в конзолата се генерира арта, по която са разположени разнообразни противници. Когато се опитаме да стъпим върху поле на противник, то героят го атакува и след това, ако противникът е все още сред живите,той отвръща с атака. Ако се опитаме да стъпим на невалидно поле се връща пояснително съобщение за “невалиден ход” и се дава възможност за повторно въвеждане. След всеки ход се изписват стандартен изход на актуална информация за картата и последиците от последния ход на играча.

2.Преглед на предметната област

За целта са реализирани следните класове:

1.Main.cpp – Клас, съдъжащ стартовото меню и генериращ обектите, които ще са необходими за реализацията на играта.

1. Hero ​– Базов абстрактен клас, който представлява основните функционалности на геория

2. Monster ​– Базов абстрактен клас, който представлява основните функционалности за противниците.

Класове наследници:

Обекти от тип Hero:

1. Warrior​ (Войн) – Онаследява член-данните на базовия си клас и притежава спобствена член-данна гняв, служеща за атаката му.

2. Mage​ (Магьосник) –Онаследява член-данните на базовия си клас и притежава спобствена член-данна мана, служеща за атаката му.

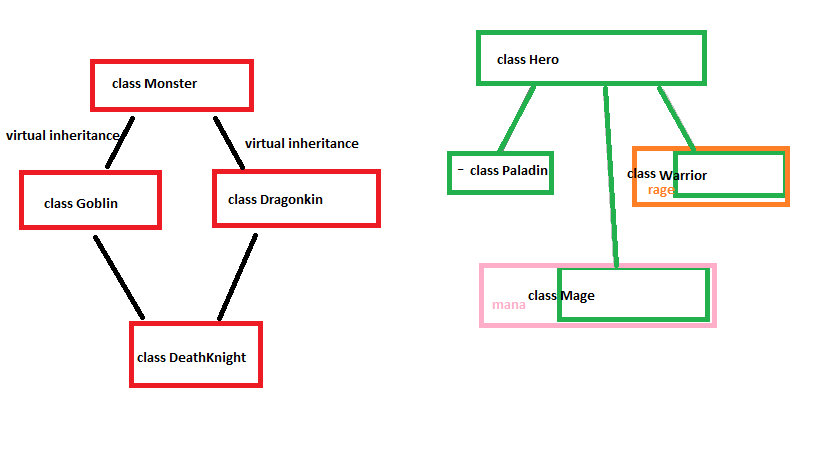
3. Paladin​ (Паладин) – Онаследява член-данните на базовия си клас и не притежава спобствени член-данни.

Обекти от тип Monster:

1. Goblin​ – Виртуално онаследява член-данните на базовия си клас

2. Dragonkin – Виртуално наследява член-данните на базовия си клас

3. Death Knight​ – Наследява обектите от тип Goblin и Dragonkin.



Сложността в този проект идва от:

1. Множествено наследяване – основен проблем тук е „Диамантеният проблем“ . Появява се заради класа DeathKnight, който е необходимо да онаследи два класа, тъй като притежава функционалностите и на Goblin, и на Dragonkin. Тогава става конфликт, поради наличието на идентични член-данни в наследниците на Monster. Проблемът се решава чрез виртуално наследяване на Monster в базовите класове на DeathKnight, т.е на class Goblin и class Dragonkin.
2. Наличие на полиморфизъм – Появява се заради различните функционалности на класовете наследници.Всяко чудовище има свой собствен тип и формула за атака и защита.Всеки герой има собствен тип,своя собствена формула за атака и защита.

3.Проектиране

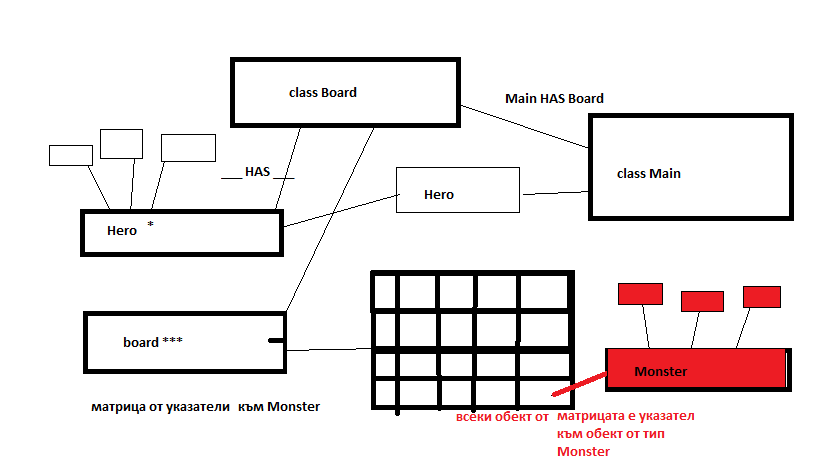
1.Концепция

Играта започва със старт меню, в което потребителят избира дали иска да създаде нов герой или да зареди от файл вече съществуващ такъв.След което потребителят избира вид и име на героя.

Генерира се карта с противници. Потребителят трябва да избере посока, в която иска да се предвижи. След въвеждане съответно на команда w a d или s , ако позицията е в обсега на картата, той се премества, в противен случай се изписва съобщение, че героят се намира извън картата и има възможност да въведе отново желана посока.

При валиден ход се проверява дали героят се е сблъскал с чудовище. Ако на текущата му позиция се намира чудовище, героят влиза в битка,която приключва в два случая: ако той умре, тогава играта приключва, ако не той може отново да избере посока, в която да се премести.

Преди всеки ход, потребителят може да избере опция да прекрати играта.

2.Архитектура

Програмата започва да се изпълнява от Main.cpp, където се намира main() функцията. Създават се обект от тип Hero и обект от тип Board. Тъй като Hero е абстрактен клас, в конструктора на Board се подава указател Hero\*.

1.Class Board()

Извиква се конструктор, на който е подаден указател към обект от тип Hero.Създава се матрица от указатели от тип Monster.

Матрицата се запълва като се генерира индекс на произволен принцип ( rand ) , като чрез switch case конструкцията се извикват на случаен принцип конструкторите на класовете наследници на Monster. Генерират се и празни полета.

След създаването на Hero и Board \*\*\*, програмата влиза в Gameloop(), член- функция на Board, която представлява същинстака част на играта.

В тялото се извикват две функции

1. drawBoard(); - чрез вложен цикъл се обхожда матрицата от указатели към Monster на конзолата се изписват типовете на съответните чудовища или празно място, ако на тази позиция указатея е nullputr. При обхождането се рисува и Hero на съответните му координати, като в началото той винаги започва от (0,0). Картата е нарисувана

2.processInput(); - тази функция представлява „динамичната“ част от играта. Изисква се посока за движение върху картата. Ходът се проверява чрез функцията isValidMove(dir) .Aко ходът не е валиден се изписва съобщение и потребителят отново трябва да въведе посока.

3.При валидна команда героят се премества, чрез член-функцията на класа Hero, Move(dir), която подава своя параметър на switch case конструкция и координатите на Hero се променят.

4. След като се променят следва проверка дали новите координати съотвтстват на място в матрицата, което не е празно, т.е е заедно от противник. Ако противник съществува следва битка.

5. Битката се осъществява чрез член-функцията на Board , Battle(). В тялото и има while цикъл, който се върти, докато героят е жив.

6. Самата битка представлява атака на герой, последвана от защина на противник, при условие че противникът е жив той се защитава и атакува обратно, ако противникът е победен, героят напуска битката, leavesBattle() се изписва подходящо съобщение и потребителят трябва да въведе ново поле, което иска да атакува.

6.Цикълът се завърта и картата се рисува отново.

**-Абстрактни базови класове**

2.Class Monster()

Съществени функции:

virtual int Attack(int = 0) = 0;

virtual void Defend(int) = 0;

virtual char getType() const = 0;

3.Class Hero()

Същетвени функции:

void Move(char);

void LevelUp()

virtual int Attack() = 0;

virtual void Defend(int) = 0;

virtual char getType() = 0;

void leavesBattle();

Функцията leavesBattle() служи за индикатор дали героите могат да използват допълнителните си сили.Тъй като повечето герои имат спецални атаки,които се активират при всяко влизане в битка,

**-Класове наследници на Monster**

**Всички наследници на Monster притежават собствена член-данна и член-функция type и getType() . Също така чисто виртуалните функции от Monster са пренаписани. Всеки наследник има уникален тип, атака и защита**

4. class Goblin : virtual public Monster

5.class Dragonkin : virtual public Monster

6. class DeathKnight : public Dragonkin , public Goblin

Специфично за него е, че наследява два класа поради способността му да редува атаките на Goblin и Dragonkin.

-**Класове наследници на Hero**

Всички наследници на Hero притежават собствена член-данна и член-функция type и getType() . Също така чисто виртуалните функции от Hero са пренаписани. Всеки наследник има уникален тип, атака и защита

7. class Warrior : public Hero

Специфичното за него е, че има собствена член-данна rage, която се генерира в защита и нападение. Също така увеличава атаката.

8. class Mage : public Hero

Специфичното за него е, че има собствена член-данна mana, която служи при атака на противник.

9. class Paladin : public Hero

Няма специални способности, но има член-данна numAttacks , която служи за брояч. Всяка трета атака е 50% по-силна.

**Насоки за в бъдеще:**

Да може да се записват героите в текстов файл.