

پروژه پیشرفته ساخت بازی مدیریت

استاد بطحائیان
محمد عرفان حمیدی مقدم



شغل : نجار (Carpenter)

کلاس ها : Tool , Nail , Woodplank, Furniture

ساب کلاس های Tool : Handsaw , Hammer

ساب کلاس های Furniture : Chair

کلاس Tool

```
// tool.hpp
#ifndef TOOL_H
#define TOOL_H

#include<iostream>
using std::string;

class Tool
{
protected:
    float durability = 0;
    string ID = "";
public:
    virtual void use();
    virtual void printInfo();
    virtual void setID(string);
    virtual string getID();
    virtual void setDurability(float);
    virtual float getDurability();
};
#endif //TOOL_H
```

دو متغیر durability و ID در حالت Protected دارد.

توابع ویرچوال برایش در حالت Public تعریف شده اند.

توابع setter و getter تعریف شده اند. توابع use و printInfo تعریف شده اند.

کلاس Handsaw

```
// handsaw.hpp
#ifndef HANDSAW_H
#define HANDSAW_H

#include "tool.hpp"

class Handsaw : public Tool
{
public:
    Handsaw(float , string);
    Handsaw(string);

    void setID(string);
    string getID();
    void setDurability(float);
    float getDurability();

    void use();
};
#endif /* HANDSAW_H */
```

کلاس Handsaw از کلاس Tool ارث برده و درون آن دو کانستراکتور تعریف شده.
Handsaw(float , string) ← هم مقدار durability و هم ID را میگیرد و مقدار دهی اولیه میکند.
Handsaw(string) ← فقط مقدار ID را میگیرد و durability را یک مقدار پیش فرض میگذارد. در این جا مقدار پیش فرض ۱۰۰ می باشد.

```
Handsaw::Handsaw(float durability, string id){
    if(durability ≥ 100)
        this->durability = 100;
    else
        this->durability = durability;
    this->ID = id;
}

Handsaw::Handsaw(string id)
{
    this->durability = 100.0;
    this->ID = id;
}
```

چک کردن مقدار گرفته

مقدار پیش فرض

تابع use کارش این است که از مقدار durability کم کند. در اینجا ۱۰ تا کم میکند. چک هم میکند که مقدار منفی نشود.

```
// handsaw.cpp
void Handsaw::use(){
    if(this->durability-10.0 <= 0)
    {
        this->durability = 0;
        std::cout << "Handsaw broke!" << std::endl;
    }
    else
        this->durability -= 10.0;
}
```

کلاس Hammer

- دقیقاً مانند کلاس Handsaw می باشد.

```
// hammer.hpp
#ifndef HAMMER_H
#define HAMMER_H

#include "tool.hpp"

class Hammer : public Tool
{
public:
    Hammer(float, string);
    Hammer(string);

    void setID(string);
    string getID();
    void setDurability(float);
    float getDurability();

    void use();
};
#endif /*HAMMER_H*/
```

```
// hammer.cpp
Hammer::Hammer(float durability, string id){
    if(durability ≥ 100)
        this->durability = 100;
    else
        this->durability = durability;
    this->ID = id;
}

Hammer::Hammer(string id){
    this->durability = 100.0;
    this->ID = id;
}

void Hammer::use(){
    if(this->durability-10.0 ≤ 0)
    {
        this->durability = 0;
        std::cout << "Hammer broke!" << std::endl;
    }
    else
        this->durability -= 10.0;
}
```

کلاس Furniture

```
// furniture.hpp
#ifndef FURNITURE_H
#define FURNITURE_H

#include <iostream>
using std::string;

class Furniture
{
protected:
    string ID;
    int price;
public:
    virtual void setPrice(int);
    virtual int getPrice();
    virtual void setID(string);
    virtual string getID();
};
#endif //FURNITURE_H
```

متغیر های ID و price در حالت Protected تعریف شده اند.
برای هر متغیر setter و getter ویرچوال در حالت Public تعریف شده است.

کلاس Chair

```
// chair.hpp
#ifndef CHAIR_H
#define CHAIR_H

#include "furniture.hpp"

class Chair : public Furniture
{
public:
    //constructors
    Chair(string);
    Chair(int , string);

    //setter getter
    void setPrice(int);
    int getPrice();
    void setID(string);
    string getID();
};
#endif //CHAIR_H
```

کلاس Chair از کلاس Furniture ارث برده و درون آن دو کانستراکتور تعریف شده.

Chair(string) ← مقدار ID را مقدار دهی اولیه می کند. مقدار price پیش فرض ۲۰ است.

Chair(int , string) ← مقدار ID و price را مقدار دهی اولیه میکند. مقدار price هم چک میکند تا منفی نشود. اگر منفی بود پیش فرض ۰ میگذارد.

```
// chair.cpp
Chair::Chair(string id){
    this->ID = id;
    this->price = 20;
}

Chair::Chair(int price, string id){
    if(price < 0)
    {
        this->price = 0;
        this->ID = id;
        std::cout << "Negative Price!" << std::endl;
    }
    else
    {
        this->price = price;
        this->ID = id;
    }
}
```

کلاس Nail

```
#ifndef NAIL_H
#define NAIL_H

enum typeNail { A , B , C , D };

class Nail{
private:
    typeNail Type;
public:
    //Constructor - Inputs : Nail type
    Nail(typeNail type);
    //Setter and Getter for nail type
    void setType(typeNail type);
    typeNail getType();
};
#endif /* NAIL_H */
```

کلاس Nail یک عضو متغیر enum در حالت Private دارد. این enum چهار حالت A B C D را دارد.
در حالت Public یک کانستراکتور که مقدار Type را مقدار دهی اولیه می کند.
getter و setter برای Type دارد.

```
Nail::Nail(typeNail type){
    Type = type;
}

//Setter and Getter for nail type
void Nail::setType(typeNail type){
    Type = type;
}
typeNail Nail::getType(){
    return Type;
}
```


کلاس Woodplank

```
#ifndef WOODPLANK_H
#define WOODPLANK_H

enum typeWood { Oak , Maple , Walnut , Teak , Ash };

class WoodPlank{
private:
    float Length; // Tool
    float Width; // Arz
    float Thickness; // Ertefa
    typeWood Type; //Type of wood

public:
    //Constructor - Inputs : Length , Width , Thickness , Type of wood
    WoodPlank(float l , float w , float th , typeWood t );

    //Setters and Getters
    void setLength(float length);
    float getLength();
    void setWidth(float width);
    float getWidth();
    void setThickness(float thickness);
    float getThickness();
    void setType(typeWood type);
    typeWood getType();
};

#endif /* WOODPLANK_H */
```

چهار متغیر در حالت Private دارد :

Length ← طول

Width ← عرض

Thickness ← ضخامت/ارتفاع

Type ← از نوع typeWood که enum است و ۵ حالت , Oak

Maple , Walnut, Teak , Ash دارد.

توابع getter و setter برای هر متغیر تعریف شده و این توابع ورودی ها را چک میکنند.

```
void WoodPlank::setLength(float length){
    if(length > 0)
        Length = length;
    else
        std::cout << "Error - Can Not Accept The Length Value" << std::endl;
}
float WoodPlank::getLength(){
    return Length;
}
void WoodPlank::setWidth(float width){
    if(width > 0)
        Width = width;
    else
        std::cout << "Error - Can Not Accept The Width Value" << std::endl;
}
float WoodPlank::getWidth(){
    return Width;
}

void WoodPlank::setThickness(float thickness){
    if(thickness > 0)
        Thickness = thickness;
    else
        std::cout << "Error - Can Not Accept The Thickness Value" << std::endl;
}
float WoodPlank::getThickness(){
    return Thickness;
}

void WoodPlank::setType(typeWood type){
    Type = type;
}
typeWood WoodPlank::getType(){
    return Type;
}
```

کانستراکتور اش چهار متغیر t , th , w , l را میگیرد و متغیر های Length , Width , Thickness , Type را مقدار دهی اولیه میکنند.

```
// woodplank.cpp
WoodPlank::WoodPlank(float l , float w , float th , typeWood t ){
    Length = l;
    Width = w;
    Thickness = th;
    Type = t;
}
```