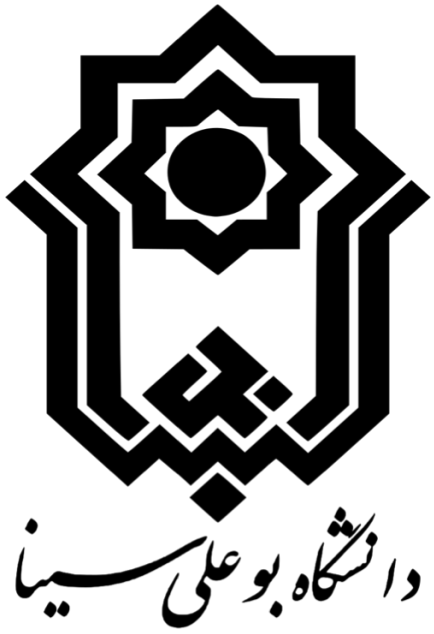
بسمه تعالی




# پروژه ترم- بازی مدیریت بیمارستان

استاد بطحاییان

محمدامین بازدار ۹۹۱۲۳۵۸۰۰۷

# شرح پروژه

پروژه ساخت بازی مدیریت بیمارستان پروژه ای در چهار فاز، به زبان سی پلاس پلاس و دارای رابط گرافیکی است. در هر فاز بخشی از پروژه تکمیل میشود.

روند کلی انجام پروژه به این صورت است که در سه فاز ابتدایی منطق برنامه به زبان سی پلاس پلاس تکمیل و در فاز آخر پروژه، رابط گرافیکی آن ساخته میشود.

# فاز اول

هدف کلی در فاز ابتدایی شکل دادن منطق کلی برنامه و پایه ریزی کلاس های ابتدایی بوده که این کلاس ها در فاز های بعدی تکمیل و بر آنها افزوده خواهد شد.

همچنین سعی شده است از تمامی ابزار خواسته شده برای انجام پروژه مانند کنتزل ورژن و سایر موارد استفاده شود.

کلاس های ساخته شده در این فاز کلاس های:

Patient-

Doctor-

Hospital-

بوده که در ادامه به مرور و توضیح آنها پرداخته خواهد شد.

# Patient.hpp

در این کلاس از داده ای شمارشی برای نوع درخواستی که بیمار دارد استفاده شده است به همین سبب دکتر با توجه به نوع بیماری آنرا درمان و به ازای هر نوع درخواست تعداد سکه ای بابت درمان گرفته شود.
بنابراین هر بیمار چهار وضعیت دارد:

- نیاز به جراحی
- نیاز به ویزیت شدن
- نیاز به قرص
- درمان شده

کانستراکتور این کلاس تنها نوع درخواست را به عنوان ورودی میگیرد.

```
class patient
{
public:
    enum class demand {SURGERY = 1, PILL, VISIT, CURED};        //patient states
    patient(patient::demand);                                   // patient constructor

    void setDemand(patient::demand);
    patient::demand getDemand() const;

    void printState();
private:
    patient::demand patientDemand;                             //patient state
};
```

# Doctor.hpp

در این کلاس تنها وضعیت دکتر در نظر گرفته شده است. از آنجا که احتمالا در رابط گرافیکی برای درمان هر بیمار مدت زمانی صرف شود بنابراین در این کلاس مشغول بودن یا نبودن دکتر در نظر گرفته شده.

```
class doctor
{
public:
    doctor() {}                     //default constructor
    ~doctor() {}                    // default destructor

    bool isBusy();
private:
    bool busy = false;
};
```

# Hospital.hpp

هدف از ایجاد این کلاس مدیریت بازی است. ایجاد شی از این کلاس سبب شروع بازی خواهد شد. برای ساخت این بیمارستان درجه سختی بازی و نام بیمارستان گرفته میشوند. با توجه به درجه سختی بازی تعداد دکتر و بودجه متفاوتی در ابتدای بازی در نظر گرفته میشود.

برخی از متد های این کلاس:

## addDoctor()

این متد برای افزودن(خرید) دکتر برای بیمارستان است. برای درجه سختی های متفاوت قیمت دکتر نیز متفاوت است. در این تابع متد decCoin() فراخوانی میشود که باتوجه به تعداد دکتر خریداری شده از مقدار سکه های بیمارستان کسر میشود.

دکتر های برنامه نیز در یک وکتور از نوع کلاس دکتر ذخیره میشوند.

## Treat()

ورودی این متد یک شی از جنس کلاس بیمار است. با فراخوانی این متد در وکتور سرچ میشود و اگر دکتری بدون بیمار بود آن بیمار به آن تخصیص داده میشود و پس از درمان با توجه به نوع بیماری تابع incCoin() و در غیر اینصورت خطا چاپ میشود.

نکته:در این فاز از اکسپشن ها نیز استفاده شده است.

کد های نوشته شده در main.cpp بی ربط به پروژه و صرفا برای تست کردن کلاس ها و متد ها قرار داده شده اند.

```cpp
class hospital
{
public:
    enum class difficulty {EASY = 1, NORMAL, HARD};        //game difficulty

    hospital(std::string, hospital::difficulty);           //constructor: name and game difficulty
    ~hospital(){ }                                         // default deatructor

    std::string getName() const;

    hospital::difficulty getDifficulty() const;

    void addDoctor(unsigned int);                         //adding doctor to hospital
    unsigned int getDoctor() const;                       // returns number of doctors

    unsigned int getCoin() const;                         // returns coin numbers

    void treat(patient&);

    void print() const;

private:
    std::string hospitalName;
    void setName(std::string);

    hospital::difficulty gameDiff;                       //game difficulty
    std::vector<doctor> doctors;                         // hospital docs

    void incCoin(unsigned int);                          // increase player coins
    void decCoin(unsigned int);                          // decrease player coins
    unsigned int coin = 0;
};
```

پایان فاز اول

# فاز دوم

هدف از این بخش افزودن کلاس و ویژگی ارث بری به پروژه بود.

در نتیجه کلاس person به پروژه افزوده شده. این کلاس ابسترکت بوده و قابلیت ایجاد شی از این کلاس وجود ندارد. برای ابسترکت قرار دادن کلاس یک تابع بی هدف قرار داده شده.

ویژگی های این کلاس نام شخص ومشغول به کار (یا درمان ) بودن شخص است. کلاس خای patient و doctor از این کلاس ارث میبرند.

```cpp
#ifndef PERSON_HPP
#define PERSON_HPP

class person
{
public:
    person(std::string = "person");
    ~person() { }

    void setName(std::string);
    std::string getName();

    bool isBusy();
    void changeState();

    virtual void makeAbstract() = 0;        // only to make the abstract class
private:
    std::string name;
    bool busy = false;
};
```

از جمله تغییرات ایجاد شده در این فاز میتوان به افزودن تخصص به کلاس doctor اشاره کرد. برای این منظور یک داده شمارشی در نظر گرفته شده که مقادیر آن پزشک جراح  پزشک عمومی است. هدف از این عمل ایجاد تطابق بر اساس نیاز بیمار و تخصص پزشک است. ممکن است با توجه به نوع پیشروی پروژه در ادامه تخصص ها و بیماری های جدید به کلاس دکتر و بیمار افزوده شود.

پایان فاز دوم

# منابع

Stackoverflow.com