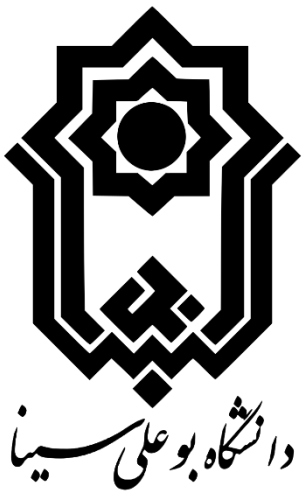
بسمه تعالی



عنوان:

«گزارش کار فاز سوم پروژه پایانی درس برنامه سازی پیشرفته»

«Sewing Game»

استاد:

سرکار خانم مهندس بطحائیان

دانشجو:

نسترن منصوری

۹۶۱۲۳۵۸۰۴۱

ترم تحصیلی:

۴۰۰۱

فهرست

هدف:

هدف از پیاده سازی این فاز استفاده از وکتورها و پیاده سازی استثناها می باشد.

توضیحات کلی کد در این فاز:

در این فاز علاوه بر تکمیل ارث بری های موجود سعی شده است تا از وکتور و استثناها هم استفاده شود. به این منظور کلاس Manto و کلاس Design تغییر پیدا کرده و کلاس جدید تحت عنوان Pants تعریف شده است. در ادامه تغییرات موجود در دو کلاس Manto و Design و پیاده سازی کلاس Pants به طور کامل توضیح داده شده است.

کلاس Manto:

```
//Manto.h definition

#include <iostream>
#include "Design.h"

#ifndef MANTO_H
#define MANTO_H


class Manto
{
    public:

        Manto();

        void set_button();

        void set_brooch();

    private:

        bool exist_button = false ;
        bool exist_brooch = false ;

};

#endif // MANTO_H
```

در header این کلاس تغییری ایجاد نشده است اما پیاده سازی کانستراکتور و دو تابع set_button و set_brooch آن به طور کامل تغییر یافته است. ابتدا به توضیح کانستراکتور می پردازیم.

کانستراکتور:

```
Manto::Manto()
{
    string choose;
    char response=' ';
    for(int i=1;i<=2;i++)
    {
        cout<<"**** Button\tOR\tBrooch ****"<<endl;
        cin>>choose;


        if ((choose=="button")||(choose=="Button")||(choose=="brooch")||(choose=="Brooch"))
        {

            if((choose=="button")||(choose=="Button"))
            {
                response='b';
            }
            if((choose=="brooch")||(choose=="Brooch"))
            {
            response='r';
            }
        }
        else
        {
            throw invalid_argument("You entered an invalid value\nSo let's go to the next step");
        }

        switch (response)
        {
        case 'b':
            set_button();
            break;
        case 'r':
            set_brooch();
            break;
        default:
            throw invalid_argument("You entered an invalid value\nSo let's go to the next step");
        }
        if(2==i)
        {
            cout<<"So let's go to the next step"<<endl;
            break;
        }
    }
}
```

همانطور که در تصویر مشخص است در قسمت اول کد یک متغیر از نوع string به نام choose و یک متغیر از نوع char به نام response تعریف شده است. هنگامی که یک شی از این کلاس ایجاد می شود و کانستراکتو رآن فراخوانی می شود، ابتدا وارد یک حلقه می شویم. حلقه ای به طول ۲؛به این منور که از کاربر پرسیده میشود که میخواهد دکه و یا سنجاق سینه داشته باشد یا خیر؟ حال کاربر باید مشخص کند که می خواهد دکمه یا سنجاق سینه داشته باشد یا خیر اگر بخواهد که باید button یا brooch را وارد کند و مقدار ورودی در choose ذخیره می شود. حال در این قسمت کامپایلر چک میکند که ورودی کاربر در صورت خواست دکمه یا سنجاق سینه درست باشد. اگر ورودی کاربر دکمه باشد مقدار 'b' در response ذخیره می شود واگر ورودی کاربر

سنجاق سینه باشد مقدار 'r' در response ذخیره می شود. اگر ورودی کاربر هیچ یک از این مقادیر نبوده باشد و درواقع choose به طور اشتباه پر شود، استثنایی پرتاب خواهد شد با این مضمون که "مقدار ورودی استباه است پس به مرحله بعد می رویم". اما اگر استثنا رخ ندهد و برنامه به صورت عادی عمل کند در قسمت بعدی با استفاده از switch...case مقدار موجود در response چک می شود تا مشخص شود که کدام یک از توابع set_button و set_brooch فراخوانی شود. در این قسمت هم اگر مقدار response هیچ یک از مقادیر مورد نظر نباشد باز هم استثنا پرتاب خواهد شد.

توابع set_button و set_brooch:

```
void Manto::set_button()
{
    cout<<"Set the Button!";
    exist_button = true;
    cout<<endl;

}

void Manto::set_brooch()
{

    cout<<"Set the Brooch!";
    exist_brooch = true;
    cout<<endl;

}
```

همانطور که در تصویر مشخص است این دو تابع نسبت به فاز قبلی دچار تغییر شده اند. تغییرات آن ها به این صورت است که با فراخوانی هریک پیغای تحت عنوان "سنجاق سینه ست شد" و "دکمه ست شد" داده می شود و متغیر های exist_brooch و exist_button از false به true تغییر خواهند یافت.

کلاس Pants:

این کلاس، کلاس شلوار است. این کلاس هم ماند=ند سایر کلاس های موجوددر پروژه از دو قسمت header و cpp. تشکیل شده است که در ادامه توضیح داده می شود.

Pants.h:

```
//Pants.h definition

#include <iostream>
#include <string>

#ifndef PANTS_H
#define PANTS_H


class Pants
{
    public:
        Pants();

        void set_pants_size();
        void set_belt();

        virtual ~Pants();


    private:
        bool exist_belt=false;
        std::string pants_size;
};

#endif // PANTS_H
```

همانطور که مشخص است این کلاس دو متغیر خصوصی دارد به نام های exist_belt که از نوع Boolean بوده و برای مشخص کردن حضور یا عدم حضور کمربند می باشد و pants_size که از نوع رشته بوده و برای ذخیره سازی بلند یا کوتاه بودن شلوار است. همچنین دو تابع به غیر از کانستراکتور آن وجود دارد به نام set_pants_size و set_belt.

Pants.cpp:

کانستراکتور:

```
#include <iostream>
#include <string>

#include "Pants.h"

using namespace std;


Pants::Pants()
{
    set_pants_size();
    set_belt();
}
```

در کانستراکتور این کلاس تنها تابع set_belt و تابع set_pants-size فراخوانی می شود.

تابع set_pants-size:

```
void Pants::set_pants_size()
{
    string pSize;
    cout<<"Choose the size you want."<<endl<<"Long OR Short?\t";
    cin>>pSize;

    if((pSize=="long")||(pSize=="Long"))
    {
        pants_size="long";
    }
    else if((pSize=="short")||(pSize=="Short"))
    {
        pants_size="short";
    }
    else
    {
        throw invalid_argument("You entered an invalid value");
    }
}
```

این تابع متغیری دارد از نوع رشته که سایز شلوار در ان نگه میدارد. از کاربر خواسته میشود که سایز مورد نظر خود را انتخاب و وارد کند. سپس مقدار ورودی در pSize ذخیره می شود. بعد چک می شود که مقدار آن long بوده یا short که اگر هریک از اینها باشد و درست نوشته شده باشد در متغیر خصوصی pants_size ذخیره می شود. اگر مقدار ورودی اشتباه باشد استثنا رخ می دهد.

تابع set_belt:

```
void Pants::set_belt()
{
    char ch;
    cout<<"Do you want your pants to have a belt?(y/n)";
    cin>>ch;
    switch(ch)
    {
    case 'y':
    case 'Y':
      exist_belt=true;
        break;
    case 'n':
    case 'N':
        cout<<"OK"<<endl;
        break;
    default:
        cout<<"Input value is incorrect!!./a"<<endl;
    }
}
```

این تابع هم متغیر از نوع کاراکتر دارد که از کاربر پرسیده می شودکه آیا میخواهد شلوارش کمربند داشته باشد یا خیر؟ سپس مقدار ورودی توسط کاربر در متغیر کاراکتری ch ذخیره می شودو بعد با استفاده از switch...case چک می شود که اگر مقدار ورودی 'y' بود مقدار متغیر خصوصی exist_belt، true می شود در غیر اینصورت مقدار پیش فرض آن که false بود باقی میماند و یک پیغام "OK" چاپ خواهد شد. اگر مقدار ورودی هیچ یک از مقادیر موردنظر نباشد هم یک استثنا رخ می دهد.

کلاس Design:

```cpp
#include <iostream>
#include <string>
#include <vector>

#include"Manto.h"
#include "Pants.h"

#ifndef DESIGN_H
#define DESIGN_H


class Design : public Manto , public Pants
{
    public:
        Design(std::string = "empty");
        void choose_design();
        void set_design(std::string);
        std::string get_design()const;
        void save_design();
        std::string get_save_design();

    private:

    std::string type_des;
    vector <std::string> selected_design ();  //save designs
};

#endif // DESIGN_H
```

Design.h:

در قسمت header این کلاس مشخص است که از دو کلاس Manto و Pants ارث میبرد. دو متغیر خصوصی دارد به نام های type_des از نوع رشته برای ذخیره نوع طراحی کاربر و selected_design از نوع وکتور برای ذخیره طراحی های انتخاب شده در طول بازی. همچنین به غیر از کانستراکتور توابع دیگری هم وجود دارد در قسمت public این کلاس که در بخش cpp. توضیح داده خواهد شد.

Design.cpp:

کانستراکتور:

```
#include <vector>

#include "Design.h"
#include "Manto.h"
#include "Pants.h"


using namespace std;

Design::Design()
{
    choose_design();
}
```

در کانستراکتور این کلاس فقط تابع choose_design فراخوانی می شود.

تابع choose_design:

```
void Design::choose_design()
{
    string str;

    cout<<"Choose the design you want:";
    cout<<"\t 1.Manto\t 2.Pants:\t";

    cin>>str;
    set_design(str);

    cout<<endl;
}
```

این تابع متغیری از نوع رشته دارد تا مقدار منتخب و ورودی کاربر از طراحی های موجود را ذخیره کند؛ سپس با این متغیر تابع set_design فراخوانی می شود.

تابع set_design:

```
void Design::set_design(string des)
{
    type_des=des;
    //selected_design.push_back(type_des);
    if((type_des=="Manto")||(type_des=="manto"))
    {
       try
       {
           Manto manto;
       }
       catch (invalid_argument &e)
       {
           cout << "Exception occurred: " << e.what() << endl;

       }
    }
    else if((type_des=="Pants")||(type_des=="pants"))
    {
       try
       {
          Pants pants;
       }
       catch (invalid_argument &e)
       {
           cout << "Exception occured: " << e.what() << endl;

       }

    }
    else
    {
        throw invalid_argument("incorrect value");
    }
    save_design();

}
```

در این تابع در ابتدا مقدار ورودی طراحی توسط کاربر در متغیر خصوصی کلاس یعنی type_des ذخیره می شود. پس چک می شود که مقدار ورودی manto است یا pants؟ اگر هیچ یک از دو مورد نبود استثنا رخ می دهد. در نهایت تابع save_design برای ذخیره سازی طراحی ها فراخوانی می شود.

تابع get_design:

```
string Design::get_design()const
{
    return type_des;
}
```

طراحی منتخب توسط کاربر را برمیگرداند.

تابع save_design:

```
void Design::save_design()
{
    string str;
    str=get_design();
    selected_design.push_back(str);
}
```

این تابع طراحی های انتخابی توسط کاربر در طول بازی را در وکتور selected_design ذخیره می کند.

تابع get_save_design:

```
string Design::get_save_design()
{
    return selected_design;
}
```

این تابع مقادیر موجود در وکتور طراحی ها را برمیگرداند.