به نام خدا

گزارش پروژه نهایی برنامه نویسی پیشرفته (زبان جاوا) .

پروژه شماره 2 (بازی)

حدیثه دارابی

بخش اول : فهم کلی روند بازی

برای شروع ، باید به درک و بررسی عناصری که در فایل پروژه درخواست شده است ، پرداخته شود .

پس فهم منطق و روش بازی لازم است . این بازی ، به نوعی شبیه سازی شده ی بازی سرنخ است . لذا برای درک بهتر، روند بردگیم سرنخ مورد بررسی قرار گرفت و تا حدودی ابهامات را برطرف کرد .

نام این بازی را به دلخواه Diamond Mystery انتخاب کردم. فرض گرفته می شود که بازیکن از طریق چیزی مانند راهنما به اسامی کارت ها آگاه است . لذا از نمایش نام آن ها در خروجی برنامه صرفه نظر شده است .

بازی برای 3 تا 6 نفر قابل اجراست که بسته به تعداد بازیکنان ، تفاوت های جزئی وجود دارد . لذا در اولین قدم از این بازی ، از کاربر پرسیده می شود که تعداد بازیکنان را در این بازی اعلام کند . در این مرحله شرطی گذاشته می شود که بررسی کند محدوده تعداد بازیکنان درست وارد شده باشد .

طبق سوال ، 3 دسته کارت داریم . کارت های اشخاص ، محل و اتاق . هر دسته را جداگانه بر میزنیم . برای این کار میتوان از متد collections.shuffle() استفاده کرد . سپس از روی هر دسته یک کارت جدا میشود و به عنوان کارت مخفی قرار می گیرد . حال همه ی کارت ها را با هم جمع کرده و بار دیگر بر می زنیم . اکنون نوبت تقسیم و توزیع کارت هاست .

از آنجایی که تعداد بازیکنان میتوانست بین 3 تا 6 نفر باشد ، بعد از جداسازی 3 کارت مخفی از 21 کارت ، 18 کارت باقی می ماند که باید به طور مساوی بین بازیکنان تقسیم شود . برای بازی 3 و 6 نفره این عمل به راحتی قابل اجراست . اما برای بازی 4 و 5 نفره ، از آنجایی که 18 کارت به 5 و 4 نفر بخش پذیر نیست ، دو راه را پیش پا می گذارد . یک راه اضافه کردن کارت به مجموع کارت هاست و راه دیگر تقسیم کارت ها به طور مساوی و قرار دادن آشکار کارت های باقی مانده روی صفحه ی بازی . بعد از پرسش ، بنا بر انتخاب راه دوم شد .

پس اگر کارتی باقی مانده باشد ، آن را چاپ می کند تا همه ی بازیکنان از آن اطلاع یابند و از لیست حدس های خود آن ها را حذف کنند .

بعد از این مرحله ، شروع کننده بازی را ، بازیکن حقیقی در نظر می گیریم . کارت هایی که در دست این بازیکن قرار دارد را نمایش می دهیم تا با آگاهی بتواند حدس های خود را اعلام کند .

اکنون که از مرحله ی توزیع کارت ها عبور کردیم ، نوبت به شروع رسمی بازی میرسد . از کاربر پرسیده میشود که کلمه ی roll را وارد کند تا بازی آغاز شود . تا زمانی که این دستور را درست وارد کند به او فرصت داده میشود . بعد از آن به صورت رندوم دو تاس ریخته میشود . نتیجه ی هر تاس و همچنین مجموع آن چاپ میشود . طبق قوانین بازی می بایست با توجه به زوج یا فرد بودن مجموع تاس های ریخته شده ، اتاقی را با ویژگی متناظر انتخاب کرد . مطابق نقشه ی خانه ، 9 اتاق داریم و هر اتاق یک عدد به خود اختصاص داده است . در دور اول بازی همه ی بازیکنان خارج از خانه قرار دارند . این نکته که بازیکن در کجا هست ، قبل از دستور تاس ریختن به او اعلام میشود .

حالا بعد از مشخص شدن مجموع تاس ها بررسی می شود که زوج است یا فرد و مطابق با آن لیستی از اتاق های زوج و یا فرد برای انتخاب ارائه میشود. (در دور اول این لیست شامل تمام زوج ها یا فرد ها (از 1 تا 9 همراه نام) میشود . اما از دور های بعدی ، اتاق های مجاور (به جز

پذیرایی) و اتاقی که بازیکن در آن بوده ، از لیست اتاق های در دسترس خارج می شود . این امکان ، سهولت انتخاب بازیکن را با توجه به قیود ذکر شده در فایل ، فراهم میکند .

اکنون می بایست بازیکن با توجه به شرایطی که دارد ، عدد دلخواه اتاق را وارد کند . (برای سهولت از وارد کردن نام اتاق چشم پوشی کرده و تنها عدد اتاق کفایت میکند!) حالا بازیکن وارد اتاق شده است . باید از او درمورد حدسش از شخصیت و محل مخفی شدن الماس سوال کرد . بعد از طرح سوال به صورت جداگانه ، بازیکن حدس خود را از هر یک به تفکیک وارد می کند . یک بار به صورت کلی حدس او را نمایش می دهیم .

حال باید از نفر بعد از او بررسی شود . چنانچه فردی کارتی مشابه حدس بازیکن اصلی دارد ، اعلام کرده و مخفیانه برای بازیکن اصلی ، تنها یک کارت را آشکار میکند و اگر اینگونه نبود ، از آن فرد عبور میکنیم . با توجه به خواست سوال ، در صورتی که یک کارت آشکار شود ، دور تمام است . در پایان دور ، دو راه پیش روی بازیکن اصلی است . اعلام حدس نهایی و یا ادامه دور با نفر بعدی . پس از او سوال می شود که آیا میخواد حدس نهایی خود را اعلام کند ؟ و او با بله یا خیر جواب میدهد . در صورتی که پاسخش مثبت باشد ، مجدد درمورد سه کارتی که فکر میکند مخفی است ، از او پرسیده می شود . پاسخ بازیکن با کارت های مخفی شده در ابتدای بازی ، مقایسه می شود . اگر منطبق بود به او تبریک گفته و او برنده ی بازی است . اگر نبود ، پاسخ های درست به او اعلام شده و بازی تمام می شود . البته این برای زمانی است که بازی در نسخه ساده ی خود قرار دارد . در نسخه ارتقا یافته ، با توجه به فایل پروژه ، بازیکنی که حدس اشتباه میزند ، از دور خارج شده و تنها میتواند اگر کارت مشابهی با حدس بازیکن دیگر داشت اعلام کند .

اما در صورتی که پاسخش منفی باشد ، یعنی نخواهد حدس نهایی خود را اعلام کند ، بازیکن های بعدی به ترتیب به صورت رندوم تاس ریخته و به اتاق مورد نظر می روند و حدس اولیه را به صورت رندوم اعلام میکنند . در نسخه ساده بازی تا همین نقطه مجاز به فعالیت هستند . در نسخه ارتقایافته اما ، میتوانند حدس نهایی را نیز به صورت رندوم اعلام کنند و از کارت های مشابه دیگر بازیکنان آگاه شوند .

بدین ترتیب بازی صورت گرفته و به پایان میرسد . در بخش های بعدی به بررسی برخی چالش ها و همچنین توضیحاتی در مورد کد ها پرداخته می شود .

بخش دوم : چالش ها و برخی توضیحات کلی

در ابتدا باید کلاس هایی را تعریف کنیم . در اولین بار نوشتن کد ، کلاس های پابلیک جداگانه برای شخصیت و محل و اتاق در نظر گرفته شد و یک کلاس هم برای روند کلی بازی تعریف شد . اما در قدم های بعدی از آنجا که یک مجموعه کارت با ویژگی ها ومتد های یکسان داریم ، تصمیم گرفتم یک کلاس اصلی کارت ها را تعریف کنم و بعد زیر کلاس های شخصیت و محل و اتاق را تشکیل دهم که از کلاس اصلی ارث بری می کنند . کلاس اصلی را انتزاعی در نظر گرفتم . چنانچه رابط در نظر گرفته میشد ، ممکن بود مجبور به تکرار برخی قسمت های کد باشم . مانند پیاده سازی متد ها . لذا به نظر رسید انتزاعی مناسب تر باشد .

در توضیحات مربوط به کد درمورد کلاس ها بیشتر گفته می شود .

از دیگر چالش ها این بود که بازیکن اصلی نمیدانست چه کارت هایی در دست دارد . پس کد به گونه ای تغییر یافت که این امکان برای بازیکن اصلی فراهم شود .

مورد دیگر ترتیب قرارگیری متد ها و ... در main برنامه بود که با جا به جایی های مناسب این چالش حل شد .

مورد دیگر این بود که بازیکن را از دور اول در یک اتاق فرض میگرفت . لذا در پیشنهاد اتاق ها دچار مشکل میشد . پس دور اول به صورت جداگانه تعریف شد تا شرایط خاصی را اعمال کند . یعنی در دور اول بازیکن خارج از خانه است . بعد از ریختن تاس ها است که به اتاق منتخب خود می رود .

چالش دیگر ، ترکیب انگلیسی و فارسی بود ورودی ها بود . لذا همه ی ورودی ها و همچنین خروجی ها به انگلیسی برگردانده شد تا نیازی به تغییر زبان سیستم پشت هم نباشد . (می توان با تغییر در برنامه ، ترجمه ی آن ها را قرار داد .)

همچنین با استفاده از lowercase مسئله ی کوچک یا بزرگ بودن حروف ، حل شدنی است .

از دیگر چالش ها در مرحله کارت ها این بود که کارت های باقی مانده برای همه آشکار نمیشد . لذا این نکته در کد لحاظ گشت .

یک مورد دیگر ، این بود که بازیکنان رندوم حدس های رندوم اولیه خود را اعلام می کنند اما برخی این حدس ها از میان کارت های در دست خودشان است . پس این کارت ها بهتر است از لیست رندوم آن ها حذف شوند .

در طول بازی برای حدس باید به این نکته توجه کرد که حدس های بازیکنان رندوم که از آن ها عبور میشود ، ممکن است یک یا دو مورد نادرست داشته باشد و تنها به صورت کلی غلط باشد . پس نمی توان همه ی احتمالات را ا بین آن ها خارج کرد .

بخش سوم : توضیحات کد

کلی:

کلاس پایه cards یک کلاس انتزاعی است که نماینده کارت‌هاست. این کلاس دارای یک ویژگی name برای نگه‌داری نام کارت است.و دلیل استفاده از این کلاس همانطور که قبل تر نیز گفته شد ، نیاز به تعریف انواع مختلف کارت‌ها (شخصیت، مکان و اتاق)، از این کلاس پایه است.

زیرکلاس CharacterCards از کلاس Cards ارث بری می‌کند و نماینده کارت‌های شخصیت است.دلیل استفاده از این کلاس عبارت است از : داشتن کارت‌های شخصیت با جزئیات خاص خودشان.

زیرکلاس PlaceCards نیز همانند قبلی است .

و در نهایت زیرکلاس RoomCards که از کلاس Cards ارث‌بری می‌کند و نماینده کارت‌های اتاق است. این کلاس دارای ویژگی roomNo نیز برای نگه‌داری شماره اتاق است و دلیل استفاده از این کلاس ،برای داشتن کارت‌های اتاق با شماره اتاق‌های منحصر به فرد ، می باشد .

حال کلاسی دیگر برای مدیریت بازیکنان و ... تعریف می شود . کلاس Players نماینده بازیکنان بازی است. این کلاس دارای ویژگی‌هایی برای نگه‌داری نام بازیکن، اتاق فعلی و قبلی بازیکن و دست کارت‌های بازیکن است. همچنین متدهای مختلفی برای تنظیم اتاق فعلی، دریافت اتاق قبلی، و حدس زدن کارت‌ها وجود دارد.تعریف این کلاس برای مدیریت اطلاعات و عملیات مربوط به هر بازیکن ، صورت گرفته شده است .

کلاس در نظر گرفته شده ی بعدی ، کلاس DiamondMystery است .این کلاس شامل متد main برای اجرای بازی است. ابتدا تعداد بازیکنان را از کاربر دریافت می‌کند.کارت‌های شخصیت، مکان و اتاق را مقداردهی اولیه و سپس ترکیب می‌کند.کارت‌های مخفی انتخاب شده و بقیه کارت‌ها بین بازیکنان تقسیم می‌شوند. و در نهایت بازی در یک حلقه تکرار می‌شود تا زمانی که یک بازیکن به درستی حدس بزند یا بازی به پایان برسد.

جزئی:

همانطور که پیش تر گفته شد ، یک کلاس انتزاعی تعریف می شود که نمی توان از آن به طور مستقیم نمونه سازی کرد و نیاز است تا توسط زیرکلاس های دیگری تکمیل شود .

یک متغیر پرایوت برای نگه داری نام کارت تعریف می شود سپس سازنده ی کلاس را که نام کارت را دریافت و تنظیم میکند تشکیل می دهیم . سپس در آن به متغیر name مقدار می دهیم .

متد بعدی get برای دسترسی و بازگردانی نام کارت است . و متدی دیگر برای دریافت جزئیات کارت و برگرداندن رشته ای از نوع و نام کارت .

حال نوبت آن است که زیرکلاس ها را تشکیل دهیم . زیرکلاس مربوط به شخصیت ها که از کلاس پایه ارث بری می کند . برای این زیرکلاس نیز سازنده را قرار داده که نام شخصیت را دریافت می کند و به سازنده کلاس پایه ارجاع میدهد . سپس فراخوانی از طریق دستور super صورت می گیرد . متد دریافت جزئیات نیز از کلاس پایه اورراید می شود .

برای زیر کلاس مربوط به کارت های محل نیز به همین گونه عمل می شود . مشابها برای زیرکلاس کارت های اتاق نیز اعمال می شود با این تفاوت که به دلیل اهمیت عدد اتاق در بازی ، یک متغیر پرایوت برای نگه داری شماره اتاق در نظر گرفته ، سازنده نام و شماره را دریافت میکند و با فراخوانی از کلاس پایه برای نام و مقدار دهی roomNo سازنده بسته می شود . یک متد get داریم برای دسترسی و برگرداندن به شماره اتاق و یک متد دیگر برای نمایش جزییات که اورراید شده است .

اکنون نوبت به تعریف کلاس player است . این کلاس برای نگه داری اطلاعات بازیکنان تعریف شده است . پس باید چند متغیر شامل نام بازیکن ، اتاق فعلی ، اتاق قبلی و همچنین کارت های در دست بازیکن ، به صورت پرایوت تعریف شود .برای هر کدام متد های لازم برای تنظیم و مقدار دهی نوشته می شود .

از متد های دیگر متدی برای حدس بازیکن و چاپ حدس او است . همچنین متدی داریم برای آشکار سازی کارت اگر حدس بازیکن درست باشد .

یک حلقه برای بررسی کارت های در دست بازیکن . در اینجا بررسی می شود که آیا کارت از نوع شخصیت است و نام کارت با حدس شخصیت مطابق است یا خیر و اگر شرایط صحیح باشند ، کارت را بر میگرداند . همین رویه برای کارت های محل و اتاق (شماره) اجرا می شود . اگر هیچ کارتی مطابقت نداشته باشد مقدار null بر میگردد.

کلاس بعدی تعریف شده کلاس اصلی برنامه است که هم نام بازی است . در این قسمت یک اسکنر ایجاد کرده تا ورودی ها را از کاربر بگیرد و آن را ثابت و استتیک تعریف می کنیم .

از این مرحله به بعد ، متد اصلی برنامه را نوشته و بررسی می کنیم .

ابتدا تعداد بازیکنان را دریافت می کنیم و سپس لیست کارت های هر دسته را مقدار دهی میکنیم . این مقدار دهی در این کد با استفاده از متد هایی صورت گرفته که بعد تر در کد مشاهده می شود . سپس هر دسته را با استفاده از متد شافل بر میزنیم . از هر دسته یک کارت با اندیس صفر را جدا کرده و به عنوان کارت های مخفی قرار می دهیم .

سپس با اضافه کردن هر دسته از اندیس 1 تا اخر دسته ، کارت ها را با هم تجمیع کرده و بعد مجدد با متد شافل بر می زنیم .

لیستی از بازیکنان ایجاد می کنیم . بازیکن حقیق مخاطب است و باقی از شما 1 به بعد با توجه به تعداد بازیکنان ، شماره گذاری می شوند . کارت ها بین بازیکنان تقسیم می شود . اگر بعد از تقسیم مساوی کارت باقی ماند ، به صورت آشکار برای همگی چاپ می شود . همچنین بعد از آن کارت های در دست بازیکن حقیقی برای او فاش می شود .

حال متغیری برای نگه داری وضعیت برد بازی ، متغیری برای نگه داری وضعیت اولین دور بازی تعریف می شود .حلقه ی بعدی تا زمانی است که بازی تمام نشده است و در آن حلقه برای نوبت هر بازیکن و بررسی اینکه بازیکن حقیقی است قرار داده شده است . اگر حقیقی بود یا نبود متد مربوط به هر کدام فراخوانی می شود . بعد از بررسی هر بازیکن، قرار می دهیم که دور اول تمام شده است . در بخش آخر بدنه اصلی کد هم چاپ إتمام بازی را در نظر می گیریم .

متد هایی را که در بدنه استفاده شد ، بررسی کنیم :

اولین متد دریافت تعداد نفرات بازیکنان بود . در این متد بعد از دریافت وردوی بررسی می شود که در بازه باشد و تعداد بازیکنان را بر میگرداند .

متد های بعدی برای مقداردهی اولیه به هر دسته کارت تعریف شده ست .

متد بعد ، متد شبیه سازی نوبت بازیکن حقیقی است . در دور اول او خارج از خانه است . در دور های بعدی اما ، در یک اتاق است . اتاق های قبلی و فعلی بررسی می شوند . از بازیکن درخواست می شود تا دستوری را برای ریختن تاس وارد کن . دستور باید به درستی وارد شود .

تاس یک و دو تعریف شده با توجه به متد مربوط ، با هم جمع شده و به کاربر نمایش داده می شوند .

شرایط اتاق های در دسترس بررسی شده و لیستی از آن ها نمایش داده می شود . اگر اتاقی در دسترس نباشد این نکته را اعلام میکند . (فکر می کنم چنین شرایطی پیش نیاید مگر با قیود خاص)

حال کاربر انتخاب می کند که به کدام اتاق برود و عدد مورد نظر را وارد می کند . اتاق فعلی او ، می شود اتاق انتخابی او . از او سوال می شود که حدسش از شخصیت و محل چیست و ورودی ها از بازیکن دریافت می شود .

اگر کسی از بازیکنان کارت مشابه داشته باشد به بازیکن اصلی نمایش داده می شود . در این صورت این دور بازی تمام است و از بازیکن اصلی پرسیده می شود که حدس نهایی میزند یا خیر . برای سهولت اگر کلمات تماما با حروف کوچک هم وارد شوند ، دستور را می پذیرد .

اگر پاسخ مثبت بود ، مقدار حدس نهایی را باز میگرداند .

حدس نهایی خود تابعی جداگانه است . در این متد به صورت جداگانه در مورد حدس نهایی کاربر از هر دسته سوال میشود و ورودی دریافت می گردد .

بررسی صورت می گیرد ، در شرط اگر مطابق بود بازیکن برنده می شود و اگر نه پاسخ های صحیح نمایش داده می شود .

متد بعدی شبیه ساز نوبت بازیکن رندوم است . تقریبا همان مراحل بازیکن حقیقی ست منتهی به صورت رندوم و با این تفاوت که او اجازه ی حدس نهایی ندارد و صرفا با تاس ریختن و ورود به اتاق مجاز و منتخب میتواند حدس اولیه داشته باشد .

متد بعدی متد تاس ریختن است که به صورت رندوم صورت می گیرد .

متد بعدی بررسی اتاق های مجاز است که با توجه به قوانین بازی طراحی شده است . اول زوج و فرد بود با توجه به زوج یا فرد بودن مجموع تاس های ریخته شده بررسی شده است . سپس اینکه اتاق نمیتواند اتاق قبلی باشد . مجاور نباید باشد (اختلاف 1 نیست !) و پذیرایی که اتاق شماره 1 هست مجاور محسوب نمی شود . در نهایت اتاق های مجاز و در دسترس بر گردانده می شوند .

و اخرین متد نیز ، متد ورود به اتاق با شماره است که در قبل نیز به آن اشارتی شده است .

بدین ترتیب نسخه ی ساده تر کد به اتمام می رسد .