دانشکده مهندسی کامپیوتر

برنامه‌سازی پیشرفته
امتحان عملی دوم
بخش برنامه‌نویسی شیء گرا

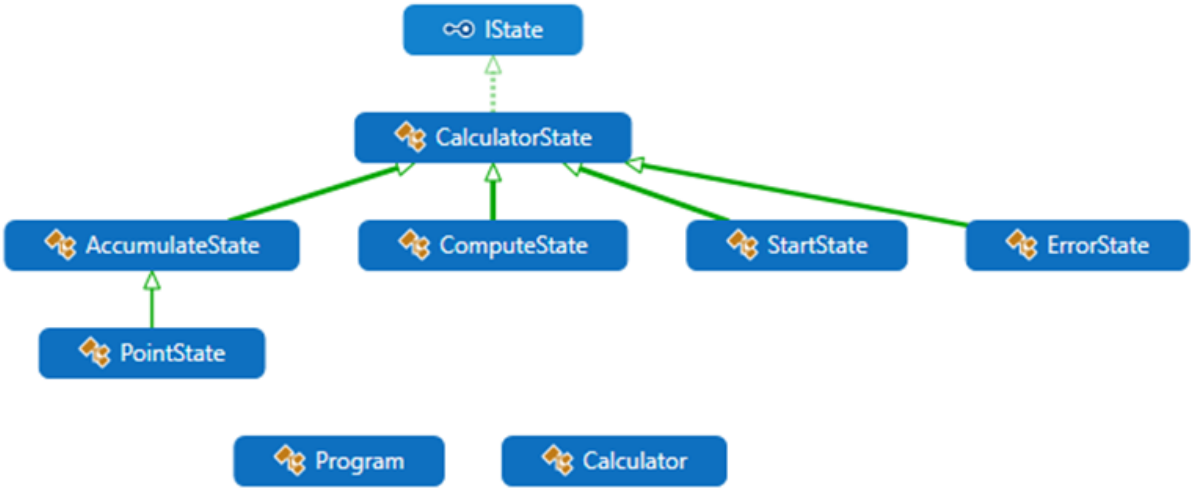
سید صالح اعتمادی *

دوم تیرماه ۱۴۰۳

هدف این بخش از امتحان آزمودن آشنایی شما با برنامه‌نویسی شیءگرا می‌باشد. در این راستا و برای حفظ یکپارچگی امتحان، سوال‌ها را بر اساس الگوی برنامه‌نویسی شیء گرا به نام الگوی حالت یا StatePattern مطرح می‌کنیم. اگر چه این الگو را جداگانه معرفی نکرده‌ایم اما این الگو بر اصول و الگوهای برنامه‌نویسی شیء گرا بنا نهاده شده و شامل هیچ مفهوم برنامه‌نویسی جدیدی نمی‌باشد.

برای آزمودن شما مقدار قابل توجهی کد آماده به شما داده شده است. از دیدن این کدها وحشت نکنید. سعی کنید با خواندن کدها هر کلاس را به تنهایی متوجه بشوید که چکار می‌کند، روابط بین کلاس‌ها را هم متوجه بشوید. ممکن است که نتوانید تصور کنید کل مجموعه تمام کلاس‌ها با هم چکار می‌کنند، ولی اگر هر کلاس را متوجه بشوید و ارتباطش با کلاس‌های دیگر را هم متوجه بشوید، می‌توانید هر کلاس را درست پیاده‌سازی کنید. علاوه بر این به اهمیت و فایده یونیت تست پی خواهید برد که چقدر به دیباگ کردن و درست کردن کد کمک می‌کند. برای تمرین می‌توانید ابتدا از یونیت تست‌ها استفاده نکنید و ببینید آیا می‌توانید سوال را حل کنید؟

*با تشکر از آقای امید میرزاجانی که در بهار ۹۹ نسخه اولیه این قالب را تهیه کردند.



شکل ۱: کلاسی دیاگرام StatePattern

موضوع این سوال نوشتن یک ماشین حساب بسیار ساده است که چهار عملگر اصلی را انجام می‌دهد. برای نحوه کار ماشین حساب، از یک ماشین حساب ساده توی موبایل یا کامپیوترتان استفاده کنید. این ماشین حساب هم مثل آن‌ها عمل می‌کند. با این تفاوت که فقط از صفحه‌کلید ورودی دریافت می‌کند نه با کلیک یا لمس. به عنوان مثال یک ماشین حساب باز کنید. دکمه صفر را چند بار فشار دهید. آیا صفحه نمایش تغییری می‌کند؟ نقطه (ممیّز) را فشار دهید. چه اتفاقی می‌افتد؟ چند بار دیگر هم فشار دهید. آیا صفحه نمایش تغییری می‌کند؟ حالا چند بار صفر را فشار دهید. آیا صفحه نمایش این دفعه فرقی می‌کند؟ چرا؟ در ابتدا ماشین حساب در وضعیت شروع (StartState) بود. ولی بعد از این‌که نقطه را فشار دادید از وضعیت شروع خارج شد و به وضعیت نقطه (PointState) وارد شد. در اینجا اتفاقات کلیدهای ورودی هستند. اگر با این دید به کد نگاه کنید، بهتر متوجه می‌شوید. کد را چندین بار مطالعه کنید اما از آن وحشت نکنید! قسمت‌هایی که نیاز به پیاده‌سازی شما دارد با علامت # و عدد کنار آن مانند: #۱ ، #۲ ، ... مشخص شده است. این ماشین حساب باید مثل یک ماشین حساب معمولی کار بکند. اگر مطمئن نیستید در یک حالتی باید چکار کند، یک ماشین حساب ساده بردارید و امتحان کنید. نمودار روابط کلاس ها در شکل ۱ موجود است.

مثل تمرین OOCalculator (A۸) که قبلا انجام دادید، دقت کنید که تعداد خطهای کد که شما باید اضافه کنید کم است. از دیباگ کردن برنامه برای فهم این که چطوری کار می‌کند استفاده کنید. اگر فقط روی جاهایی که کد پاک شده تمرکز کنید کارتان سخت خواهد شد. باید همه کلاس‌ها و روابطشان را متوجه بشوید تا بتونید جاهای خالی را پر کنید. بعد از اینکه از درست کار کردن برنامه اطمینان پیدا کردید، عملگر توان را به ماشین حساب اضافه کنید. نشانه عملگر توان کاراکتر ^ است.

۱ تست Zero

بعد از درست کردن پروژه‌ها و اضافه کردن فایل‌ها این تست بدون هیچ تغییری از طرف شما، پاس می‌شود. متن این تست و بخصوص متد RunTest را با دقت مطالعه کرده و خط به خط اجرا/دیباگ کنید و از تسلط بر نحوه انجام تست اطمینان حاصل کنید. به طور خلاص هدف این تست این است که بعد از باز کردن ماشین‌حساب هر چند بار دکمه صفر فشار داده شود صفحه نمایش عدد صفر را نشان داده و تغییری نمی‌کند.

۲ تست Accumulation

برای پیدا کردن شهود نسبت به هدف این تست برنامه calc.exe و یا یک ماشین‌حساب دستی باز کنید. هر چند بار عدد صفر را بزنید، آیا تغییری در صفحه نمایش مشاهده می‌کنید؟ حال، ابتدا دکمه یک را زده و بعد دکمه صفر را به دفعات فشار دهید. آیا متوجه تفاوت رفتار می‌شوید؟ برای پاس شدن این تست لازم است از دیباگر استفاده کرده و تغییرات لازم را در کد ایجاد کرده تا تست پاس بشود.

۳ تست AccumulateState

هدف این تست مانند بخش قبل است. با این تفاوت که وقتی ماشین‌حساب اجرا می‌شود، هر عددی غیر از صفر باید رفتاری متفاوت از عدد صفر داشته باشد. ولی همانطور که در تست قبل مشاهده کردید، این رفتار متفاوت فقط در ابتدا است. بعد از اینکه عددی غیر از صفر وارد شد، دیگر عدد صفر با دیگر اعداد تفاوتی نمی‌کند.

۴ تست PointState

هدف این تست،آزمون وارد کردن درست اعداد اعشاری در ماشین حساب می‌باشد.

۵ تست PointsOnlyState

برای پاس شدن این تست لازم است از دیباگر استفاده کرده و تست را دیباگ کنید. این تست زمانی پاس می‌شود که توی ماشین‌حساب هر چند بار دکمه نقطه فشار داده شود، فقط یک صفر و یک نقطه روی صفحه نمایش نشان داده شود. برای نمونه و مقایسه می‌توانید از یک ماشین‌حساب دستی یا برنامه calc.exe استفاده کنید.

۶ تست ExtraPoint

مجددا برای فهمیدن هدف این تست از یک ماشین‌حساب استفاده کنید. آیا بعد از اینکه مثلا عدد یک و یک دهم را وارد کردید فشار دادن دکمه نقطه تغییری در ماشین‌حساب ایجاد می‌کند؟ این نشان‌دهنده این است که بعد از وارد کردن نقطه، ماشین‌حساب به حالتی وارد می‌شود که وارد کردن نقاط بعدی صفحه نمایش را تغییر نمی‌دهد. مجددا با کمک گرفتن از دیباگر تغییرات لازم در برنامه را انجام دهید.

۷ تست StartState

حال که اعداد به درستی به ماشین‌حساب وارد می‌شوند، لازم است که بتوانیم محاسبات بین اعداد را انجام دهیم. برای گام اول،این تست زمانی پاس می‌شود که اگر دکمه بعلاوه فشار داده شد، ماشین‌حساب به حالت ComputeState تغییر حالت دهد.

۸ تست Sum

بعد از این مقدمات، لازم است که ماشین حساب یک حساب ساده را بتواند انجام دهد. به این معنی که بعد از فشار دادن دکمه مساوی، نتیجه محاسبه نمایش داده شود.

۹ تست ErrorState

چنانچه بعد از نمایش نتیجه یک محاسبه، دکمه مساوی مجددا فشار داده شود، لازم است حالت ماشین‌حساب به ErrorState تغییر پیدا کند و صفحه نمایش تغییری نکند.

۱۰ تست Multiply

حال که عملگر جمع به درستی پیاده‌سازی شد، نوبت عملگر ضرب می‌باشد. تغییرات لازم برای پاس شدن این تست را اعمال کنید.

۱۱ تست MultipleSum

چنانچه تست‌های قبل به درستی پیاده‌سازی شده باشند، این تست نیز بدون هیچ تغییری باید پاس بشود.

۱۲ تست Divide

این تست درستی اجرای عملگر تقسیم را راست‌آزمایی می‌کند.

۱۳ تست StartingPoint

چنانچه تست‌های قبل به درستی پیاده‌سازی شده باشند، این تست نیز بدون هیچ تغییری باید پاس بشود.

۱۴ تست Power

حال که چهار عمل اصلی را برای ماشین‌حساب پیاده سازی کردید، نوبت پیاده‌سازی یک عملگر جدید می‌باشد. تغییرات لازم را برای پاس شدن تست توان اعمال کنید.