اشکان شکیبا (9931030)

سوال اول

الف)

Polymorphism یا چندریختی، تکنیکی برای انجام اعمالی تقریبا مشابه و هم نام بر روی داده هایی ست که همگی از نوعی کلاس مشخص ارث بری میکنند. برای مثال میتوان کلاسی مانند Animal تعریف کرد که شامل متدی برای شبیه سازی رفتار جانور است و سپس با تعریف کلاس هایی که نمایانگر جانوران مختلف هستند و از Animal ارث بری میکنند، این متد را با توجه به ویژگی های مورد نیاز، Override کرد و سپس با پیمایش بر روی کالکشنی از Animal ها، این متد را فراخوانی نمود و این متد بنابر نوع کلاس تعریف شده (Dynamic Type)، رفتار مشخصی را بروز می دهد، و در حالیکه همگی داده ها اعضایی از کالکشنی از Animal ها هستند، رفتاری چند ریختی را با توجه به ماهیت اصلی خود به نمایش میگذارند.

Substitution زمانی رخ میدهد که در رفرنسی با Static Type کلاس والد، آدرس داده ای از کلاس فرزند (Dynamic Type) ذخیره کنیم. برای مثال اگر کلاس Bot از کلاس Player ارث بری کند، با اجرای ;()Player player=new Bot، این پدیده رخ میدهد و داده ای با Static Type: Player و Dynamic Type: Bot ذخیره میشود.

Abstract class کلاسی ست که بخشی از بدنه آن نوشته نشده است و همه کلاس هایی که از این کلاس ارث بری میکنند ملزم به تکمیل و Override این متد ها هستند. متد های abstract بدنه ای ندارند اما همه کلاس های فرزند باید بدنه ای را برای هر یک از این متد ها پیاده سازی کنند.

Interface چیزی تقریبا مشابه کلاس است، با این تفاوت که نمیتوان متدی را در آن پیاده سازی کرد و به نوعی قراردادی برای تعیین ویژگی های والد است. اصلی ترین مزیت Interface، ایجاد قابلیت ارث بری از چند مفهوم است که با کلاس ها قابل انجام نیست.

ب)

1) نادرست، در صورت استفاده از چنین چیزی، برنامه کامپایل نمیشود و از نظر معنایی نیز مفهومی ندارد.

2) درست

3) درست

4) درست

5) نادرست، متد private در هیچ کلاس دیگری، از جمله کلاس های فرزند، قابل دسترسی نیست و از این رو چنین چیزی ممکن نیست.

6) درست

ج)

Overloading به معنای تفاوت در پارامتر های ورودی متد یا کانستراکتور است که این تفاوت میتواند در تعداد، نوع و یا ترتیب پارامتر ها ظاهر شود.

در واقع در Overloading یک متد یا کانستراکتور، این متد یا کانستراکتور با signature های متفاوت در یک کلاس نوشته میشود.

اما Overriding، به معنای بازنویسی متد یا کانستراکتور پیش تر نوشته شده در کلاس والد است که در کلاس فرزند صورت میگیرد و حتی میتواند شامل فراخوانی متد یا کانستراکتور نوشته شده در کلاس والد (با کلیدواژه super) نیز باشد.

همچنین در Overriding، همواره signature متد یا کانستراکتور مورد نظر باید در هر دو کلاس والد و فرزند یکسان باشد (بدین معنی که نام متد یا کانستراکتور و تعداد، نوع و ترتیب پارامتر های ورودی آنها یکسان باشد)

سوال دوم

به طور کلی، داده ای با Dynamic Type کلاس فرزند را میتوان در رفرنسی با Static Type کلاس والد ذخیره کرد، اما عکس این موضوع شدنی نیست و منجر به خطای کامپایل میشود؛ بنابراین...

الف) خطوط اول و سوم مجاز هستند، زیرا هر Husky یک Animal و هر Bulldog یک Mammal است؛ اما خطوط دوم و چهارم کامپایل نمیشوند زیرا هر Mammal لزوما یک Cow نیست و هر Bird نیز لزوما یک Parrot نیست!

ب،ج) با توجه به اصل تعریف شده در ابتدای پاسخ، همه دستورات مجاز هستند!

بررسی خطوط به ترتیب:

هر Whale یک Mammal نیز هست.

هر Husky یک Mammal نیز هست.

هر Dog یک Dog است!

هر Bird یک Animal نیز هست.

هر Husky یک Animal نیز هست.

هر Whale یک Animal نیز هست.

هر Husky یک Animal نیز هست.

هر Husky یک Mammal نیز هست.

هر Cow یک Animal نیز هست.

بنابراین همه دستورات مجاز هستند!

سوال سوم

برنامه نوشته شده در پوشه Q3 قرار دارد.