۱- تست کارکردی و غیرکارکردی را بیشتر تشریح کنید.

تست کارکردی یکی از رایج ترین انواع تست سامانه‌های نرم‌افزاری است که **برای بررسی صحت عملکرد** آنها استفاده می‌گردد مانند تست واحد و تست یکپارچه سازی

تست غیرکارکردی به عنوان یکی از انواع تست نرم‌افزار شناخته می‌شود که **برای بررسی جنبه‌های غیرکارکردی** (Performance)یک اپلیکیشن نرم‌افزاری تعریف شده است. این تست طراحی شده است تا آمادگی سیستم را برای بررسی پارامترهای غیرکارکردی که هیچگاه توسط تست Functional تحت آزمون قرار نمی‌گیرند، تست کرده و بررسی نماید.

یک نمونه عالی از تست غیرکارکردی بررسی این موضوع است که چه تعداد از افراد(کاربران) می‌توانند به طور همزمان به نرم‌افزار لاگین کنند.

تست غیرکارکردی به اندازه تست کارکردی اهمیت داشته و بر رضایت مشتری تاثیرگذار است.

۲–چند مورد از functional testing و non-functional testing های معروف را توضیح دهید و مثال بزنید.

از انواع پرکاربرد تست کارکردی میتوان به موارد زیر اشاره کرد :

تست یونیت(Unit Testing)

تست یکپارچه‌سازی(Integration Testing)

تست دود (Smoke Testing)

تست پذیرش کاربر(UAT-User Acceptance Testing)

*تست کارکردی **برای بررسی صحت عملکرد نرم افزار** بکار میرود.

از انواع پرکاربرد تست غیرکارکردی میتوان به موارد زیر اشاره کرد :

تست یونیت(Unit Testing)

تست کارایی(Performance Testing)

تست تحمل(Endurance Testing)

تست بار(Load Testing)

*تست غیرکارکردی **برای بررسی جنبه‌های غیرکارکردی نرم افزار** بکار میرود.

- در مورد تست دود هم سوالی شده بود، باید بگم که، اصطلاح آزمایش دود (smock test) از مهندسی الکترونیک گرفته شده است. در الکترونیک آزمایش دود یک روش آزمایش ابتدایی است که روی تجهیزات الکترونیکی انجام می شود. روش کار بدین صورت است که وسیله موردنظر به برق وصل می شود و تعمیرکار به دنبال دود، جرقه یا هر نشانه غیرعادی دیگر می گردد تا محل عیب را پیدا کند.از این اصطلاح در سایر زمینه ها مانند مدیریت نرم افزار، برای بررسی اولیه برای یافتن اشتباهات پیش پا افتاده و احمقانه استفاده می شود. در واقع تاکید می شود ابتدا این مشکلات ابتدایی برطرف و سپس کارکرد وسیله یا نرم افزار آزموده شود.

۳–در محصولی که داریم تولید میکنیم، کدام تست ها کاربردی تر است؟

به نظر من تست های واحد و یکپارچه سازی

۴–استفاده از test case های مختلف و پارادوکس آفت کش را با مثال توضیح دهید.

پاردوکس آفت‌کش‌ها که سومین اصل از اصول تست است، بیان میکند که استفاده مداوم از یک ترکیب آفت‌کش برای از بین بردن حشرات در حین کشاورزی، در طول زمان منجر به ایجاد حشرات مقاوم در برابر آفت‌کش مذبور می‌شود. بنابراین آفت‌کش‌ها بر آن حشرات ناکارآمد می‌شوند. همین امر در مورد تست‌های نرم‌افزاری نیز صادق است. اگر یک مجموعه از تست‌های تکراری مرتب انجام شوند، متد شما برای کشف نواقص جدید بی‌فایده خواهد بود. پس باید از موارد تست متفاوتی برای یک تست استفاده کرد تا از این پارادوکس در امان باشیم.

۵– کدام تست ها را میتوان بصورت خودکار انجام داد؟

در بخشی که در مورد انواع تست ها حرف زدیم اشاره کردم که این لیست کامل نیست چرا که در زمانی که این مطلب آماده میشد گفته شده بود بیش از ۱۶۰ تست داریم و هر روز هم به تعدادشون اضافه میشه اما برای نمونه چند مورد را ذکر میکنم:

تست واحد

تست یکپارچه سازی

تست بار

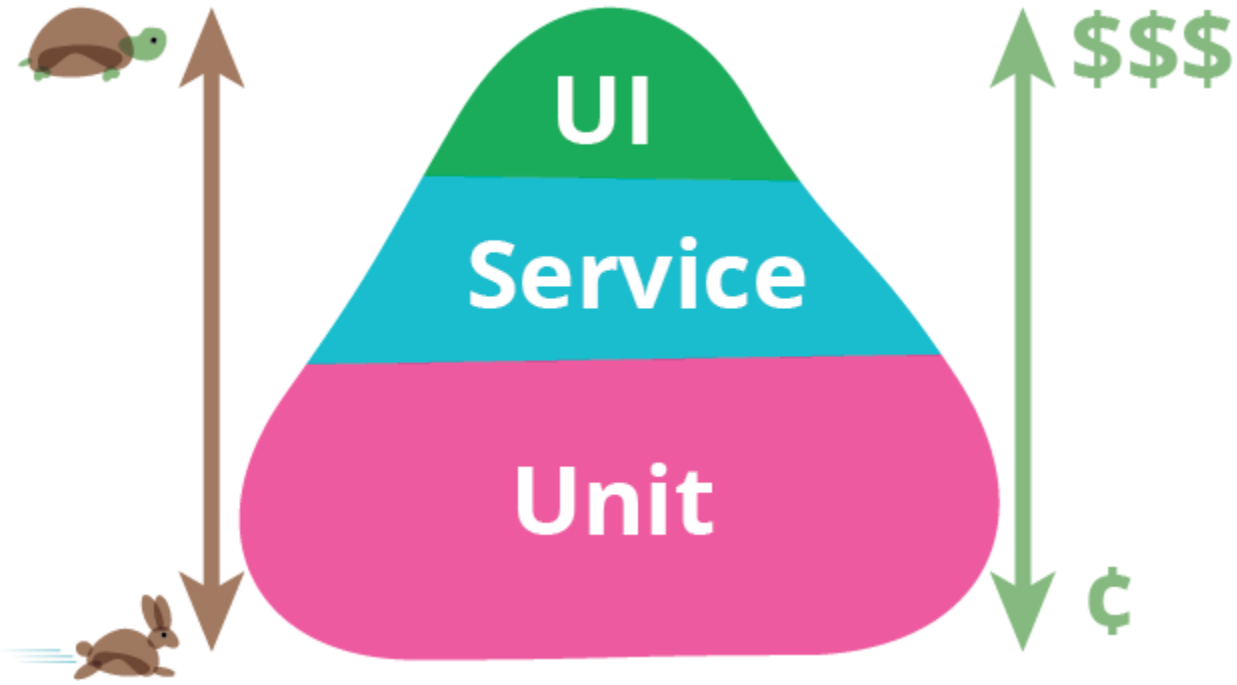
تست عملکرد

تست فشار

۶– هرم تست به چه صورتی است؟

برای خودکارسازی تست نرم‌افزار می‌توان انواع متفاوتی از تست را به کار گرفت. از تست‌های سطح پایین که برای تست قطعه کدهای نرم‌افزار نوشته می‌شوند، (Unit Test)؛ تا تست‌هایی سطح بالا که از واسط کاربری شروع می‌شوند و تست را به شکلی اجرا می‌کنند که گویا یک کاربر واقعی در حال کار با سیستم است.(End-To-End Test) .

اما سوال کلیدی این است که وقت گذاشتن برای کدامیک از این انواع تست، نتیجه‌ی بهتری را عاید ما می‌کند؟ «هرم تست» پاسخی برای همین پرسش است.



همانطور که در شکل پیداست، بهتر این است که حجم تست‌های واحد (Unit Test) بسیار بیشتر از تست‌هایی باشد که از سطح UIاجرا می‌شوند.