窗体顶端

# 专业英语

您现在的位置：[希赛网](http://www.educity.cn/" \o "希赛网) > [云阅读](http://www.educity.cn/jiaocheng/) > [软件设计师考试试题分类精解（2018版）](http://www.educity.cn/jiaocheng/zt251.html" \t "http://www.educity.cn/jiaocheng/_blank) > **试题1(2017年下半年试题71-75)**

第 13 章：专业英语作者：[希赛软考学院](http://www.educity.cn/jiaocheng/a%cf%a3%c8%fc%c8%ed%bf%bc%d1%a7%d4%ba.html" \o "希赛软考学院" \t "http://www.educity.cn/jiaocheng/_blank)    来源：希赛软考学院    2017年11月21日

# **试题1(2017年下半年试题71-75)**

The development of the Semantic Web proceeds in steps, each step building a layer on top of another. The pragmatic justification for this approach is that it is easier to achieve（  ）on small steps, whereas it is much harder to get everyone on board if too much is attempted. Usually there are several research groups moving in different directions; this（  ）of ideas is a major driving force for scientific progress. However，from an engineering perspective there is a need to standardize. So, if most researchers agree on certain issues and disagree on others, it makes sense to fix the points of agreement. This way, even if the more ambitious research efforts should fail， there will be at least（  ）positive outcomes.  
Once a（  ）has been established ，many more groups and companies will adopt it，instead of waiting to see which of the alternative research lines will be successful in the end. The nature of the Semantic Web is such that companies and single users must build tools， add content， and use that content. We cannot wait until the full Semantic Web vision materializes-it may take another ten years for it to be realized to its full（  ）(as envisioned today, of course).（71）A．conflicts  
B.consensus  
C.success  
D.disagreement  
  
（72）A．competition  
B.agreement  
C.cooperation  
D.collaboration  
  
（73）A．total  
B.complete  
C.partial  
D.entire  
  
（74）A．technology  
B.standard  
C.pattern  
D.model  
  
（75）A．area  
B.goal  
C.object  
D.extent

**试题分析**

语义网络的发展是一步一步的，每一步都建立在另一部的基础之上。选择这种方法的现实理由就是因为很容易对一小步取得成功，而如果想要一步到位就难得多。通常，很多研究组织都是从不同方向考虑的，这种思想的竞争的方式是科学进步的驱动力。然而，从工程的角度来说是需要进行标准化的。因此，如果大多数研究者同意某个观点不同意另一个的时候，改正观点是有意义的。这样，即使再宏大的研究努力也会失败，可能会有局部的积极效果。  
一旦一个技术被建立，许多组织和企业都会采纳，而不是等待并查看其他研究线是否会获得成果。语义网络的性质就是让企业和单个用户必须构建工具，添加内容并使用。我们不会等着整个语义网络被物化——因为实现它的全部内容需要再过十年时间（当然是按照今天所设想）。

**试题答案**

（71）C（72）A（73）C（74）A（75）D

# **试题2(2017年上半年试题71-75)**

The beauty of software is in its function，in its internal structure，and in the way in which it is created by a team. To a user，a program with just the right features presented through an intuitive and（  ）interface is beautiful.To a software designer，an internal structure that is partitioned in a simple and intuitive manner，and that minimizes internal coupling is beautiful.To developers and managers ，a motivated team of developers making significant progress every week，and producing defect-free code，is beautiful.There is beauty on all these levels.  
our world needs software--lots of software. Fifty years ago software was something that ran in a few big and expensive machines. Thirty years ago it was something that ran in most companies and industrial settings. Now there is software running in our cell phones，watches，appliances，automobiles，toys，and tools. And need for new and better software never（  ）.As our civilization grows and expands，as developing nations build their infrastructures，as developed nations strive to achieve ever greater efficiencies，the need for more and more Software（  ）to increase. It would be a great shame if，in all that software，there was no beauty.  
We know that software can be ugly. We know that it can be hard to use，unreliable ，and carelessly structured. We know that there are software systems whose tangled and careless internal structures make them expensive and difficult to change. We know that there are software systems that present their features through an awkward and cumbersome interface. We know that there are software systems that crash and misbehave. These are（  ）systems. Unfortunately，as a profession，software developers tend to create more ugly systems than beautiful ones.  
There is a secret that the best software developers know. Beauty is cheaper than ugliness. Beauty is faster than ugliness. A beautiful software system can be built and maintained in less time，and for less money ,than an ugly one. Novice software developers don't. understand this. They think that they have to do everything fast and quick. They think that beauty is（  ） .No! By doing things fast and quick，they make messes that make the software stiff，and hard to understand，Beautiful systems e flexible and easy to understand. Building them and maintaining them is a joy. It is ugliness that is impractical.Ugliness will slow you down and make your software expensive and brittle. Beautiful systems cost the least build and maintain，and are delivered soonest.  
  
（71）A． Simple  
B. Hard  
C. Complex  
D. duplicated  
  
（72）A．happens  
B.exists  
C.stops  
D.starts  
  
（73）A． starts  
B.continues  
C.appears  
D.stops  
  
（74）A．practical  
B.useful  
C.beautiful  
D.ugly  
  
（75）A．impractical  
B.perfect  
C.time-wasting  
D.practical

**试题分析**

软件的优点在于其功能，内部结构以及由团队创建的方式。对于用户来说，通过直观和\_\_\_\_界面呈现的正确功能的程序是美丽的。对于软件设计师来说，分割的内部结构是一种简单而直观的方式，最小化内部耦合是美观的。对于开发人员和经理来说，一个积极的开发团队每周都取得重大进展，并且生产无缺陷的代码是美丽的。所有这些级别都有美丽。  
我们的世界需要大量软件。五十年前，软件是在大多数公司和工业环境中运行的。现在软件存在在我们的手机，手表，电器，汽车，玩具和工具中。并且对新的和更好的软件的需求永远不会\_\_\_\_。随着我们文明的发展和壮大，随着发展中国家建设基础设施，发达国家努力实现更高的效率，越来越多的软件需求\_\_\_\_增长。如果在所有的软件中没有美丽的话，这将是一个很大的耻辱。  
我们知道软件可能是丑的。我们知道它可能很难使用，不可靠，粗心大意的结构。我们知道有一些软件系统的纠结和粗心的内部结构使得它们变得昂贵和难以改变。我们知道有一些软件系统通过尴尬和繁琐的界面来呈现其功能。我们知道有软件系统崩溃和行为不端。这些都是\_\_\_\_系统。不幸的是，作为一个专业，软件开发人员倾向于创建丑陋的系统比美丽的系统更多。  
这是最好的软件开发者知道的秘密。美丽的比丑陋的更便宜。美丽的比丑陋的更快。一个美丽的软件系统相当于一个丑陋的系统来说，建立和维护要花的时间与金钱会少得多。很多新手软件开发人员不明白这一点。他们认为做每一个事情必须快速，更快速。他们认为美是\_\_\_\_。没有！通过快速，快速地做事情，他们使软件变得僵硬，难以理解。美观的系统灵活易懂。建立和维护他们是一种快乐。丑陋是不切实际的。丑陋会减慢你的速度，会使你的软件昂贵而脆弱。美观的系统成本最低，建立和维护成本最低，交货时间最短。  
71 A simple（简单） B hard（困难） C complex（复杂） D duplicated（被复制）  
72 A happens（发生） B exists（存在） C stops（停止） D starts（开始）   
73 A starts （开始） B continues（持续） C appears（出现） D stops（停止）   
74 A practical （实用的）B useful（有用的） C beautiful（美丽的） D ugly（丑陋的）   
75 A impractical （不实用的）B perfect（完美的） C time-wasting（浪费时间） D practical（实用的）  
  
71：A  
解析：The beauty of software is in it's function, in it's internal structure, and in the way in which it is created by a team. To a user, a program with just the right features presented through an intuitive and simple interface, is beautiful.  
72：C  
解析：And need for new and better software never stops.  
73：B  
解析：As our civilization grows and expands, as developing nations build their infrastructures, as developed nations strive to achieve ever greater efficiencies, the need for more and more software continues to increase.  
74：D  
解析：These are ugly systems  
75: A  
解析：They think that beauty is impractical

**试题答案**

（71）A（72）C（73）B（74）D（75）A

# **试题3(2016年下半年试题71-75)**

      Software entities are more complex for their size than perhaps any other human construct, because no two parts are alike (at least above the statement level). If they are, we make the two similar parts into one, a（  ）, open or closed.  In this respect software systems differ profoundly from computers,buildings, or automobiles, where repeated elements abound.  
       Digital computers are themselves more complex than most things people build; they have very large numbers of states. This makes conceiving, describing, and testing them hard. Software systems have orders of magnitude more （  ）than computers do.  
         Likewise, a scaling-up of a software entity is not merely a repetition of the same elements in  larger size; it is necessarily an increase in the number of different elements. In most cases, the elements interact with each other in some（  ）fashion,and the complexity of the whole increases much more than linearly.  
         The complexity of software is a(an)（  ）property, not an accidental one. Hence descriptions of a software entity that abstract away its complexity often abstract away its essence.Mathematics and the physical sciences made great strides for three centuries by constructing simplified models of complex phenomena, deriving properties from the models, and verifying those properties experimentally. This worked because the complexities（  ）in the models were not the essential properties of the phenomena. It does not work when the complexities are the essence.  
          Many of the classical problems of developing software products derive from this essential  complexity and its nonlinear increases with size. Not only technical problems but management  problems as well come from the complexity.  
（71）A．task  
B.job  
C.subroutine  
D.program   
  
（72）A．states  
B.parts  
C.conditions  
D.expressions   
  
（73）A．linear  
B.nonlinear  
C.parallel  
D.additive   
  
（74）A．surface  
B.outside  
C.exterior  
D.essential   
  
（75）A．fixed  
B.included  
C.ignored  
D.stabilized

**试题分析**

软件实体的尺寸比任何其他人类构造更复杂，因为没有两个部分相同（至少在语句级上）。如果是，我们将两个相似的部分分成一个，一个（71），开放或关闭。在这方面，软件系统与计算机，建筑物或汽车有着深刻的区别，其中重复的元素很多。  
数字电脑本身比大多数人理解的很多情况都要更复杂。这使得构思，描述和测试他们非常复杂。软件系统比计算机更多（72）数量级。  
同样地，软件实体的放大不仅仅是较大尺寸的相同元素的重复;必然增加不同要素的数量。在大多数情况下，这些元素以（73）的方式彼此相互作用，并且整体的复杂性比线性增加更多。  
软件的复杂性是（74）的属性，而不是偶然的。因此，消除其复杂性的软件实体的描述往往会抽象出其本质。数学和物理科学通过构建复杂现象的简化模型，从模型中导出属性，并通过实验验证这些属性，在三个世纪以来取得了长足的进步。这是因为模型中的复杂性（75）不是现象的基本属性。当复杂性是本质时，它不起作用。  
开发软件产品的许多经典问题源于这一重要的复杂性，其非线性随着尺寸而增加。不仅技术问题，管理问题也来自于复杂性。

**试题答案**

（71）C（72）A（73）B（74）D（75）C

# **试题4(2016年上半年试题71-75)**

In the fields of physical security and information security, access control is the selective restriction of access to a place or other resource. The act of accessing may mean consuming, entering, or using. Permission to access a resource is called authorization （授权）．

An access control mechanism（  ）between a user (or a process executing on behalf of a user) and system resources, such as applications, operating systems, firewalls, routers, files, and databases. The system must first authenticate（验证）a user seeking access. Typically the authentication function determines whether the user is（  ）to access the system at all. Then the access control function determines if the specific requested access by this user is permitted. A security administrator maintains an authorization database that specifies what type of access to which resources is allowed for this user. The access control function consults this database to determine whether to（  ）access. An auditing function monitors and keeps a record of user accesses to system resources.

In practice, a number of（  ）may cooperatively share the access control function. All operating systems have at least a rudimentary（基本的）, and in many cases a quite robust, access control component. Add-on security packages can add to the（  ）access control capabilities of the OS. Particular applications or utilities, such as a database management system, also incorporate access control functions. External devices, such as firewalls, can also provide access control services.

（71）A．cooperates  
B.coordinates  
C.connects  
D.mediates  
  
（72）A．denied  
B.permitted  
C.prohibited  
D.rejected  
  
（73）A．open  
B.monitor  
C.grant  
D.seek  
  
（74）A．components  
B.users  
C.mechanisms  
D.algorithms  
  
（75）A．remote  
B.native  
C.controlled  
D.automated

**试题分析**

在物理安全和信息安全领域，访问控制是选择性地限制访问某个地方或其他资源。访问行为可能意味着消耗，进入或使用。授权访问资源称为授权。  
访问控制机制介于用户（或代表用户执行的进程）和系统资源之间。资源如应用程序、操作系统、防火墙、路由器、文件和数据库。系统必须先验证（验证）寻求访问权限的用户。通常，认证功能确定用户是否能被允许访问该系统。然后，访问控制功能确定该用户的特定请求的访问是否被允许。一个安全管理员维护一个授权数据库，该数据库指定该用户允许哪些资源的访问类型。访问控制功能查询此数据库以确定是否授权访问。审计功能监控和保存用户对系统资源的访问记录。  
实际上，很多组件可以协同共享访问控制功能。所有的操作系统至少有一个基本的，在许多情况下是一个非常强大的访问控制组件。附加安全软件包可以添加到操作系统的本地安全控制功能。特定的应用程序或实用程序，如数据库管理系统，还包括访问控制功能。外部设备（如防火墙）也可以提供访问控制服务。

**试题答案**

（71）D（72）B（73）C（74）A（75）B

# **试题5(2015年下半年试题71-75)**

In a world where it seems we already have too much to do, and too many things to think about, it seems the last thing we need is something new that we have to learn.

But use cases do solve a problem with requirements:with（  ）declarative requirements it's hard to describe steps and sequences of events.

Use cases, stated simply, allow description of sequences of events that, taken together, lead to a system doing something useful.As simple as this sounds,this is important. When confronted only with a pile of requiements, it's often（  ）to make sense of what the authors of the requirements really wanted the system to do.In the preceding example, use cases reduce the ambiguity of the requirements by specifying exactly when and under what conditions certain behavior occurs;as such, the sequence of the behaviors can be regarded as a requirement. Use cases are particularly well suited to capture approaches. Although this may sound simple, the fact is that（  ）requirement capture approaches, with their emphasis on declarative requirements and "shall" statements,completely fail to capture fail to capture the（  ）of the system's behavior. Use cases are a simple yet powerful way to express the behavior of the system in way that all stakeholders can easily understand.

But,like anything, use cases come with their own problems, and as useful as they are,they can be（  ）.The result is something that is  as bad, if not worse, that the original problem.Therein it's important to utilize use cases effectively without creating a greater problem than the one you started with.

（71）A．plenty    
B.loose    
C.extra    
D.strict   
  
（72）A．impossible    
B.possible    
C.sensible    
D.practical   
  
（73）A．modern    
B.conventional    
C.different    
D.formal   
  
（74）A．statics    
B.nature    
C.dynamics    
D.originals   
  
（75）A．misapplied    
B.applied    
C.used    
D.powerful

**试题分析**

在这个世界上，似乎我们有太多的事情要去做，有太多的事情要去思考，那么需要做的最后一件事就是必须学习新事物。  
而用例恰恰可以解决带有需求的问题：如果具有严格声明的需求，则很难描述事件的步骤和序列。  
简单地说，用例可以将事件序列的说明放在一起，引导系统完成有用的任务。正如听起来一样简单---这很重要。在面对很多需求的时候，通常不太可能理解需求的作者真正想要系统做什么。在前面的例子中，通过指定特定行为发生的时间和条件，用例减少了需求的不确定性。这样的话，行为的顺序就可以当作是一种需求。用例特别适用于捕捉这类需求。尽管听起来可能很简单，但事实情况是由于常规的需求捕捉方法所侧重的是声明需求和“应该怎么样”的陈述，因此完全无法捕捉系统行为的动态方面。用例是一种简单而有效的表达系统行为的方式，使用这种方式所有参与者都很容易理解。  
但是与任何事物一样，用例也存在自己的问题---在用例非常有用的同时，人们也可能误用它，结果就产生了比原来更为糟糕的问题。因此重点在于：如何有效地使用用例，而又不会产生比原来更严重的问题。

**试题答案**

（71）D（72）A（73）B（74）C（75）A

# **试题6(2015年上半年试题71-75)**

Why Have Formal Documents?

Finally, writing the decisions down is essential. Only when one writes do the gaps appear and the（  ）protrude(突出).The act of writing turns out to require hundreds of mini-decisions,and it is the existence of these that distinguishes clear,exact policies from fuzzy ones.

 Second.the documents will communicate the decisions to others. The manager will be continually amazed that policies he took for common knowledge are totally unknown by some member of his team . Since his fundamental job is to keep everybody going in the（  ）directon, his chief daily task will be communication, not decision-making,and his documents will immensely（  ）this load.

Finally,a manager,s documents give him a data base and checklist. By reviewing them（  ）he sees where he is, and he sees what changes of emphasis or shifts in direction are needed.

The task of the manager is to develop a plan and then to realize it. But only the written plan is precise and communicable. Such a plan consists of documents on what,when, how much,where,and who.This small set of critical documents（  ）much of the manager's work. If their comprehensive and critical nature is recognized in the beginning, the manager can approach them as friendly tools rather than annoying busywork. He will set his direction much more crisply and quickly by doing so.

（71）A．inconsistencies   
B.consistencies   
C.steadiness   
D.adaptability   
  
（72）A．other   
B.different   
C.another   
D.same   
  
（73）A．extend   
B.broaden   
C.lighten   
D.release   
  
（74）A．periodically   
B.occasionally   
C.infrequently   
D.rarely   
  
（75）A．decides   
B.encapsulates   
C.realizes   
D.recognizes

**试题分析**

为什么要有正式的文档？

首先，将决策写下来是关键的。只有写出后差距才能出现，矛盾才能突出。写的过程是需求成百上千的小决策的过程，这些的存在将清楚的、准确的政策从模糊的政策中区分出来。

其次，文档能够作为同其他人的沟通渠道。项目经理将会不断感到惊奇的是，许多理应被普遍认同的测量，全然不为团队的一些成员所知。既然他的基本工作是使每个人在一个方向上前进，他的主要工作就是交流，而不是决策制定，他的文档能很好的减轻这个负担。

最后，项目经理的文档给他提供了一个数据库和检验表。通过定期回顾他能知道自己所处的位置，并看到为需要对重点改变什么或方向作什么变动。  
项目经理的任务是制定计划，并根据计划实现。但是只有书面计划是精确和可以沟通的。计划中包括了时间、地点、人物、做什么、资金。这些少量的关键文档封装了一些项目经理的工作。如果一开始就认识到他们的普遍性和重要性，那么就可以将文档作为工具友好地利用起来，而不会让它成为令人厌烦的繁重任务。通过遵循文档开展工作，项目经理能更清晰和快速地设定自己的方向。

**试题答案**

（71）A（72）D（73）C（74）A（75）B

# **试题7(2014年下半年试题71-75)**

Teams are required for most engineering projects. Although some small hardware or software products can be developed by individuals, the scale and complexity of modem systems is such, and the demand for short schedules so great, that it is no longer（  ）for one person to do most engineering jobs. Systems development is a team（  ）,and the effectiveness of the team largely determines the（  ）of the engineering.

    Development teams often behave much like baseball or basketball teams. Even though they may have multiple specialties, allthe members work toward（  ）.However,on systems maintenance and enhancement teams, the engineers often work relatively independently, much like wrestling and track teams.

     A team is（  ）just a group of people who happen to work together.  Teamwork takes practice and it involves special skills. Teams require common processes; they need agreed-upon goals; and they need effective guidance and leadership. The methods for guiding and leading such teams are well known, but they are not obvious.

（71）A．convenient  
B.existing  
C.practical  
D.real  
  
（72）A．activity  
B.job  
C.process  
D.application  
  
（73）A．size  
B.quality  
C.scale  
D.complexity  
  
（74）A．multiple objectives  
B.different objectives  
C.a single objective  
D.independent objectives  
  
（75）A．relatively  
B./  
C.only  
D.more than

**试题分析**

大多数工程项目需要团队完成。虽然有些小规模的硬件或软件产品可以由个人完成，但是现代系统的规模大、复杂性高以及开发周期短的极高需求，使得一个人完成大多工程工作已经不再现实。系统开发是一个团队活动，团队的效率很大程度上决定工程的质量。  
开发团队经常表现的像是棒球队或篮球队，即使棒球队或篮球队可能有多种不同专长，但是所有的队员都朝着一个目标努力。然而在系统维护和挣钱团队，工程师们的工作就像摔跤和田径队一样经常相对独立。  
团队不仅仅是一群人碰巧在一起工作，团队工作需要实践，涉及到多种特殊的技能。团队需要共同的过程，需要达成一致的目标，需要有效地指导和领导。尽管指导和领导这样的团队的方法是众所周知的，但是它们并不明显。

**试题答案**

（71）C（72）A（73）B（74）C（75）D

# **试题8(2014年上半年试题71-75)**

Cloud computing is a phrase used to describe a variety of computing concepts that involve a large number of computers（  ）through a real-time communication network such as the Internet. In science, cloud computing is a（  ）for distributed computing over a network, and means the（  ）to run a program or application on many connected computers at the same time.  
The architecture of a cloud is developed at three layers: infrastructure, platform, and application, The infrastructure layer is built with virtualized computer, storage, and network resources. The platform layer is for general-purpose and repeated usage of the collection of software resources. The application layer is formed with a collection of all needed software modules for SaaS applications. The infrastructure layer serves as the（  ）for building the platform layer of the cloud. In turn, the platform layer is a foundation for implementing the（  ）layer for SaaS applications.

（71）A．connected  
B.imlemented  
C.optimized  
D.Virtualized  
  
（72）A．replacement  
B.switch  
C.substitute  
D.synonym(同义词)  
  
（73）A．ability  
B.applroach  
C.function  
D.method  
  
（74）A．network  
B.foundation  
C.software  
D.hardware  
  
（75）A．resoruce  
B.service  
C.application  
D.software

**试题分析**

云计算是用来描述各种计算概念的短语，包括大量计算机通过网络相互连接以实现分布计算，意思是同时在很多互联的计算机上运行程序或应用的能力。 云的架构分为基础设施层、平台层和应用层三层。基础设施层由虚拟计算、存储和网络资源构成。平台层用于一组软件资源重复使用的通用目的。应用层由一组所需的软件模块构成即软件即服务（SaaS)。基础设施层作为构建平台层的基础。相反，平台层是应用层的基础，为SaaS应用实现应用层。

**试题答案**

（71）A（72）D（73）A（74）B（75）C

# **试题9(2013年下半年试题71-75)**

There is nothing in this world constant but inconstancy．--SWIFT

Project after project designs a set of algorithms and then plunges into construction of customer-deliverable software on a schedule that demands delivery of the first thing built．

In most projects,the first system built is（  ）usable,It may be too slow,too big,awkward to use,or all three．There is no （  ）but to start again,smarting but smarter,and build a redesigned version in which these problems are solved．The discard and（  ）may be done in one lump,or it may be done piece-by-piece．But all large-system experience shows that it will be done．Where a new system concept or new technology is used,one has to build a system to throw away,for even the best planning is not so omniscient（全知的）as to get it right the first time．

The management question,therefore ,is not whether to build a pilot system and throw it away．You will do that．The only question is whether to plan in advance to build a（  ）,or to promise to deliver the throwaway to customers．Seen this way,the answer is much clearer．Delivering that throwaway to customers buys time,but it does so only at the（  ）of agony（极大痛苦）for the user,distraction for the builders while they do the redesign,and a bad reputation for the product that best redesign will find hard to live down．

Hence plan to throw one away;you will,anyhow．

（71）A．almost   
B.often   
C.usually   
D.barely   
  
（72）A．alternative   
B.need   
C.possibility   
D.solution   
  
（73）A．design   
B.redesign   
C.plan   
D.build   
  
（74）A．throwaway   
B.system   
C.software   
D.product   
  
（75）A．worth   
B.value   
C.cost   
D.invaluable

**试题分析**

不变只是愿望，变化才是永恒。—SWIFT  
一个接一个的软件项目都是一开始设计算法，然后将算法应用到待发布的软件中，接着根据时间进度把第一次开发的产品发布给客户。  
对于大多数项目，第一个开发的系统并不适用。它可能太慢、太大、难以使用，或者三者兼有。要解决所有的问题，除了重新开始以外，没有其他的办法—即开发一个更灵巧或者更好的系统。系统的丢弃和重新设计可以一步完成，也可以一块块地实现。所有大型系统的经验都显示，这是必须完成的步骤。而且，新的系统概念或新技术会不断出现，因此开发的系统必须被抛弃，但即使是最优秀的项目计划也不能无所不知地在最开始就解决这些问题。  
因此，管理上的问题不再是“是否构建一个实验性的系统，然后抛弃它”，你必须这样做。现在的问题是“是否预先计划抛弃原型的开发，或者是否将该原型发布给用户”。从这个角度看待问题，答案更加清晰。将原型发布给用户，虽然可以获得时间，但是其代价高昂—对于用户，使用极其痛苦；对于重新开发的人员，分散了精力；对于产品，影响了声誉，即使是最好的再设计也难以挽回名声。  
因此，为舍弃而计划，无论如何，你一定要这样做。

**试题答案**

（71）D（72）A（73）B（74）A（75）C

# **试题10(2013年上半年试题71-75)**

So it is today. Schedule disaster,functional misfits,and system bugs all arise because the left hand doesn't know what the right hand is doing. As work（  ） ,the several teams slowly change the functions,size,and speeds of their own programs,and the explicitly or implicitly（  ）their assumptions about the inputs available and the uses to be made of outputs.  For example ,the implementer of a program-overlaying function may run into problems and reduce speed -relying on statistics that show how（  ）this function will arise in application programs. Meanwhile,back at the ranch, his neighbor may be designing a major part of the supervisor so that it critically depends upon the speed of this function. This change in speed itself becomes a major specification change , and it needs to be proclaimed abroad and weighed from a system point of view. How,then,shall teams（  ）with one another? In as many ways as possible. Informally. Good telephone service and a clear definition of intergroup dependencies will encourage the hundreds of calls upon which common interpretation of written documents depends. Meetings. Regular project meetings,with one team after another giving technical briefings , are（  ）. Hundreds of minor misunderstangings get smoked out this way. Workbook.A formal project workbook must be started at the beginning.

（71）A．starts   
B.proceeds   
C.stops   
D.speeds   
  
（72）A．change   
B.proceeds   
C.smooth   
D.hide   
  
（73）A．frequently   
B.usually   
C.commonly   
D.rarely   
  
（74）A．work   
B.program   
C.communicate   
D.talk   
  
（75）A．worthless   
B.valueless   
C.useless   
D.invaluable

**试题分析**

之所以有今天。时间表灾难功能的不适应加上系统错误，这些都是由于左手不知道右手在做什么。当工作进行的时候，一些团队慢慢改变他们自己程序的功能、大小合数独胆，并且直接或间接地改变关于输入效用的假设和由输出组成的使用。  
例如，一个覆盖程序函数的实现者可能遇上问题并且减少依赖于展现这个函数在应用程序中多么罕见的统计的速度。与此同时回到农场，他的邻居可能是设计的一个主要部分的监督人，这样它极度取决于函数的速度。这种变化在速度本身成为一个主要规划变化，它需要对外宣布，从概念系统来做衡量。  
那么，团队应当用尽可能多的方式彼此交流。  
非正式的。良好的电话服务和明确定义的组间依赖关系将鼓励成百上千的书面文件共同翻译上所依赖的调用。  
会议。定期项目会议，一个接一个给技术简报的团队是无价的。许多小的误解在这种方式下得到化解。  
工作薄。一个正式的项目工作薄必须一开始就准备。

**试题答案**

（71）B（72）A（73）D（74）C（75）D

# **试题11(2012年下半年试题71-75)**

Computers will become more advanced and they will also become easier to use．Improved speed recognition will make the operation of a computer easier．Mrtual reality（虚拟现实），the technology of（  ）with a computer using all of the human senses， will also contribute to better human and computer（  ）.Other， exotic（奇异的）models of computation are being developed， including biological computing that uses living organisms， molecular computing that uses molecules with particular（  ），and computing that uses DNA， the basic unit of heredity（遗传）to store data and carry out operations.These are examples of possible future computational platforms that， so far， are limited in abilities or are strictly（  ）.Scientists investigate them because of the physical limitations of miniaturizing circuits embedded in silicon.There are also（  ）related to heat generated by even the tiniest of transistors.

（71）A．interact   
B.interacting   
C.communicate   
D.using   
  
（72）A．interfaces   
B.behavior   
C.similarities   
D.comparison   
  
（73）A．software   
B.properties   
C.programs   
D.hardware   
  
（74）A．empirical   
B.real   
C.practical   
D.theoretical   
  
（75）A．developments   
B.advantages   
C.limitations   
D.improvements

**试题分析**

计算机将变得更加先进，也将变得更容易使用。提高了处理速度将使计算机变得更加容易操作。虚拟现实是一种用人的感官与计算机交换的技术， 也将更有利于人和计算机接口。另外，奇异的模型正在开发中，包括使用生活有机体的生物计算，使用具有特定性能的分子计算和使用DNA（遗传的基本单位）来存储数据和进行操作等技术。这些可能是未来计算平台的例子，到目前为止，计算机的能力有限，并且局限于严格的理论。科学家们也正是因为小型化的嵌入式芯片电路的物理限制而研究新的技术。也有晶体管产生热量方面的限制，哪怕是最小的晶体管。

**试题答案**

（71）B（72）A（73）B（74）D（75）C

# **试题12(2012年上半年试题71-75)**

At a basic level, cloud computing is simply a means of delivering IT resources as（  ）. Almost all IT resources can be delivered as a cloud service: applications, compute power, storage capacity, networking, Programming tools, even communication services and collaboration（  ）.

Cloud computing began as large-scale Internet service providers such as Google, Amazon, and others built out their infrastructure. An architecture emerged: massively scaled,（  ）distributed system resources, abstracted as virtual IT services and managed as continuously configured, pooled resources. In this architecture, the data is mostly resident on（  ）"somewhere on the Internet" and the application runs on both the "cloud servers" and the user's browser.

Both clouds and grids are built to scale horizontally very efficiently. Both are built to withstand failures of（  ）elements or nodes. Both are charged on a per-use basis. But while grids typically process batch jobs, with a defined start and end point, cloud services can be continuous. What's more, clouds expand the types of resources available - file storage, databases, and Web services - and extend the applicability to Web and enterprise applications.

（71）A．hardware   
B.computers   
C.services   
D.software   
  
（72）A．computers   
B.disks   
C.machines   
D.tools   
  
（73）A．horizontally   
B.vertically   
C.inclined   
D.decreasingly   
  
（74）A．clients   
B.middleware   
C.servers   
D.hard disk   
  
（75）A．entire   
B.individual   
C.general   
D.separate

**试题分析**

    基本上，云计算仅仅意味着将IT资源作为服务来提供。几乎所有的IT资源都可以作为一个云服务来提供：如应用程序，计算能力，存储容量，网络，编程工具，甚至通讯服务和协作工具。  
    开始大规模提供云计算的互联网服务提供商有谷歌，亚马逊和其他一些基础建设商。云架构的特点是：系统不断扩大，水平分布系统资源，抽象的虚拟服务和管理，然后不断配置，汇集资源。在这种体系结构中，数据主要存放在“互联网某处”的服务器和“云服务”与客户端运行的应用程序上。  
    要想建立一个能收取使用费的云网格，必须要建立经得起单个元素或节点失败的云网格。当用网格来处理一批日常工作时，需要定义一个明确的开始和结束点，云服务可以是连续的。但更重要的是，云可以扩展可用的文件存储，数据库，网络服务来适用于网络和企业应用。

**试题答案**

（71）C（72）D（73）A（74）C（75）B

# **试题13(2011年下半年试题71-75)**

 Extreme Programming (XP) is a discipline of software development with（  ）of simplicity， communication， feedback and courage. Successful software development is a team effort - not just the development team， but the larger team consisting of customer， management and developers. XP is a simple process that brings these people together and helps them to succeed together. XP is aimed primarily at object-oriented projects using teams of a dozen or fewer programmers in one location. The principles of XP apply to any（  ）project that needs to deliver quality software rapidly and flexibly.  
An XP project needs a（  ）customer to provide guidance. Customers， programmers， managers， are all working（  ） to build the system that’s needed. Customers - those who have software that needs to be developed - willlearn simple， effective ways to（  ）what they need， to be sure that they are getting what they need， and to steer the project to success.

（71）A．importance  
B.keys  
C.roles  
D.values  
  
（72）A．small-sized  
B.moderately-sized  
C.large-sized  
D.huge-sized  
  
（73）A．part-time  
B.casual  
C.seldom  
D.full-time  
  
（74）A．together  
B.by themselves  
C.separately  
D.alone  
  
（75）A．tell  
B.know  
C.communicate  
D.feedback

**试题分析**

极限编程是一个专业软件开发方法，它包含简单，沟通，反馈和勇气四大价值观。成功的软件开发是一个团队努力的结果，而这个团队不只是开发团队，而是由客户，管理和开发人员一起组成的更大的团队。极限编程是一个简单的过程，为了成功而将一些人聚集在一起，它主要是针对十几个或更少人的项目团队的面向对象开发。原则上极限编程适用于任何需要迅速和灵活提供高质量的轻量级项目。  
一个极限编程项目需要客户的全程指导，而客户、程序员和项目经理都是必须的人员。客户是指那些有软件需要被急切发展的人，我们需要和他们建立有效的沟通方式来确定他们的需求，引导项目走向成功。

**试题答案**

（71）D（72）B（73）D（74）A（75）C

# **试题14(2011年上半年试题71-75)**

Ravi， like many project（  ），had studied the waterfall model of software development as the primary software life-cycle（  ）.He was all set to use it for an upcoming project， his first assignment． However， Ravi found that the waterfall model could not be used because the customer wanted the software delivered in stages， something that implied that the system had to be delivered and built in（  ）and not as（  ）.  
The situation in many other projects is not very different. The real world rarely presents a problem in which a standard process， or the process used in a previous project， is the best choice．To be the most suitable， an existing process must be（  ）to the new problem．A development process， even after tailoring， generally cannot handle change requests．To accommodate change requests without losing control of the project， you must supplement the development process with a requirement change management process．

（71）A．customers   
B.managers   
C.users   
D.administrators   
  
（72）A．activity   
B.procedure   
C.process   
D.progress   
  
（73）A．parts   
B.modules   
C.software   
D.a whole   
  
（74）A．parts   
B.modules   
C.software   
D.a whole   
  
（75）A．modified   
B.used   
C.suited   
D.tailored

**试题分析**

    Ravi就像很多研究过以瀑布模型为软件生命周期过程的软件开发项目经理一样，他被安排使用瀑布模型去开发一个即将启动的项目，而且这是他的第一个任务。然而，Ravi发现不能在项目中使用瀑布模型，因为客户想要该软件分阶段交付，而不是作为一个整体交付。  
    在很多其他的项目中也有类似的情况，现实生活中，本来就很少有能完全按标准来进行处理的问题，可能某标准处理前一个问题非常合适，但处理现在这个问题就不定适合了。最合适的方法就是对一个新的问题必须采用切合它自身的方法。为了适应变化的变更请求而不失去对项目的控制，你必须要支持项目的发展过程与一个需求变更管理过程。

**试题答案**

（71）B（72）C（73）A（74）D（75）D

# **试题15(2010年下半年试题71-75)**

People are indulging in an illusion whenever they find themselves explaining at a cocktail(鸡尾酒)party，say， that the are “in computers，”or“ in telecommunications，”or “in electronic funds transfer”. The implication is that they are part of the high-tech world. Just between us，they usually aren’t. The researchers who made fundamental breakthroughs in those areas are in a high-tech business. The rest of us are（  ）of their work. We use computers and other new technology components to develop our products or to organize our affairs. Because we go about this work in teams and projects and other tightly knit working group（紧密联系在一起的工作小组），we are mostly in the human communication business. Our successes stem from good human interactions by all participants in the effort，and our failures stem from poor human interactions.  
　　The main reason we tend to focus on the（  ）rather than the human side of work is not because it’s more（  ），but because it’s easier to do. Getting the new disk drive installed is positively trivial compared to figurine out why Horace is in a blue funk(恐惧）or why Susan is dissatisfied with the company aver only a few months. Human interactions are complicated and never very crisp（干脆的，干净利落的）and clean in　their effects， but they matter more than any other aspect of the work.  
If you find yourself concentrating on the（  ）rather than the（  ）.you’re like the vaudeville character （杂耍人物）who loses his Keys on a dark street and looks for them on the adjacent street because，as he explains，“The light is better there!”

（71）A．creators   
B.innovators   
C.appliers   
D.inventors   
  
（72）A．technical   
B.classical   
C.social   
D.societal   
  
（73）A．trivial   
B.crucial   
C.minor   
D.insignificant   
  
（74）A．technology   
B.sociology   
C.physiology   
D.astronomy   
  
（75）A．technology   
B.sociology   
C.physiology   
D.astronomy

**试题分析**

    人们总幻觉自己参加了一个“计算机”、“电信”或“电子金融”方面的鸡尾酒会，梦想自己参与了这些高科技领域。事实是研究者们在那些高科技领域取得了根本性的突破，而其余的人都只是应用他们的研究成果而已。  
我们使用电脑和其他新技术开发各种新产品能增强我们工作团队的合作，方便大家沟通。在人类的很多事业中，成功来源于所有参与者的共同努力，失败是因为大家不相互协作。很多失败的主要原因是我们过多关注技术而忽略了人性的一面，这并不是因为人性更重要，而是因为它更容易存在问题。  
    在短短几个月内，相比于获知为什么霍勒斯心中忐忑不安或者为什么苏珊是不满意的，开发新磁盘的安装驱动要容易实现得多，这是因为人类的交流是非常复杂的，而且从来就不干脆，总是不清不楚。但它的作用确实非常重要的。  
    如果你发现自己专注于技术而不是社会学。就如一个杂耍人物在黑暗的街道丢失了钥匙，而在临近的街道去寻找，而他的原因是：“临近街道的灯光更好”。

**试题答案**

（71）C（72）A（73）B（74）A（75）B

# **试题16(2010年上半年试题71-75)**

Observe that for the programmer， as for the chef， the urgency of the patron（顾客）may govern the scheduled completion of the task， but it cannot govern the actual completion.An omelette（煎鸡蛋）， promised in two minutes， may appear to be progressing nicely.Butwhen it has not set in two minutes， the customer has two choices—waits or eats it raw.Software customers have had（  ）choices.  
　　Now I do not think software（  ）have less inherent courage and firmness thanchefs， nor than other engineering managers.But false（  ）to match the patron’s desireddate is much more common in our discipline than elsewhere in engineering.It is very（  ）to make a vigorous， plausible， and job risking defense of an estimate that is derived by no quantitative method， supported by little data， and certified chiefly by the hunches of the managers.  
　　Clearly two solutions are needed.We need to develop and publicize productivity figures， bug-incidence figures， estimating rules， and so on.The whole profession can only profit from（  ）such data.Until estimating is on a sounder basis， individual managers will need to stiffen their backbones and defend their estimates with the assurance that their poor hunches are better than wish derived estimates.

（71）A．no  
B.the same  
C.other  
D.lots of  
  
（72）A．Testers  
B.constructors  
C.managers  
D.architects  
  
（73）A．Tasks  
B.jobs  
C.Works  
D.scheduling  
  
（74）A．easy  
B.difficult  
C.simple  
D.painless  
  
（75）A．sharing  
B.excluding  
C.omitting  
D.ignoring

**试题分析**

观察一下编程人员，你可能会发现，同厨师一样，某项任务的计划进度。可能受限于顾客要求的紧迫程度，但紧迫程度无法控制实际的完成情况。就像约好在两分钟内完成一个煎蛋，看上去可能进行得非常好。但当它无法在两分钟内完成时，顾客只能选择等待或者生吃煎蛋。软件顾客的情况类似。  
　　我现在并不认为软件经理内在的勇气和坚持不如厨师，或者不如其工程理经理。但为了满足顾客期望的日期而造成的不合理进度安排，在软件领域中却比其他的任何工程领域要普遍得多。而且，非量化方法的采用，少得可怜的数据支特，加上完全借助软件经理的直觉，这样的方式很难生产出健壮可靠和规避风险的估计。  
　　显然我们需要两种解决方案。开发并推行生产率图表、缺陷率，估算规则等，整个组织最终会从这些数据的共享上获益。或者在基于可靠基础的估算出现之前，项目经理需要挺直腰杆并坚持他们的估计，确信自己的经验和直觉总比从期望得出的估计要强得多。

**试题答案**

（71）B（72）C（73）D（74）B（75）A