2022知识点练习->上午->12.专业英语

1.One is that of a software engineer and the other is a DevOps engineer. The biggest different is in their ( ). Software engineers focus on how well the computer software fits the needs of the client while a DevOps engineer has a broader focus that includes software development, how the software is deployed and providing ( ) support through the cloud while the software is continually ( ).
A software engineer creates computer programs for people to use based upon their security and function ability needs. A DevOps engineer also works on computer applications, but manages the building, deployment and operation as a( ) autormated process. Software engineers often work separately from the operations side of a business. They create the software a business client needs and then monitor the performance of their software products to determine if up grades are necessary or if more serious improvements are needed. DevOps engineers work with the operational side of a business and manage the workflow to ( ) software to smoothly function with automated processes. Both professions require knowledge of Computer programming languages.

问题 1：	A.focus	B.process	C.goal	D.function
问题 2：	A.developing	B.deploying	C.training	D.operational
问题 3：	A.developed	B.functional	C.constructed	D.secure
问题 4：	A.single	B.whole	C.continuous	D.independent
问题 5：	A.develop	B.integrate	C.analyse	D.maintain

2.Designing object -oriented software is hard,and designing（ ）object -oriented software is even harder.You must find pertinent(相关的)objects,factor them into class at the right granularity,define class interfaces and inheritances,and establish key relationships among them.You design should be specific to the problem at hand but also（ ）enough to address future problems and requirements.You also want to avoid redesign,or at least minimize it.Experienced object -oriented designers will tell you that a reusable and flexible design is difficult if not impossible to get "right" the first time.Before a design is finished,they usually try to reuse it several times,modifying it each time. Yet experienced object-oriented designers do make good designs.Meanwhile new designers are（ ）by the options available and tend to fall back on non-object-oriented techniques they've used before.lt takes a long time for novices to learn what good object-oriented design is all about.Experienced designers evidently know something inexperienced ones don't.What is it?
One thing expert designers know not to do is solve every problem from first principles.Rather, they reuse solutions that have worked for them in the past.When they find a good（ ）.They use it again and again.Such experience is part of what makes them experts.Consequently,you'll find（ ）patterns of classes and communicating objects in many object-oriented systems.

问题 1：	A.runnable	B.right	C.reusable	D.pertinent
问题 2：	A.clear	B.general	C.personalized	D.customized
问题 3：	A.excited	B.shocken	C.surprised	D.overwhelmend
问题 4：	A.tool	B.component	C.system	D.solution
问题 5：	A.recurring	B.right	C.experienced	D.past

3.Regardless of how well designed, constructed, and tested a system or application may be, errors or bugs will inevitably occur. Once a system has been（　）,it enters operations and support.
Systems support is the ongoing technical support for user, as well as the maintenance required to fix any errors, omissions,or new requirements that may arise. Before an information system can be（　）, it must be in operation. System operation is the day-to-day, week-to-week, month-to-month, and year-t-year（　）of an information system's business processes and application programs.
Unlike systems analysis, design, and implementation, systems support cannot sensibly be（　）into actual phases that a support project must perform. Rather, systems support consists of four ongoing activities that are program maintenance, system recovery, technical support, and system enhancement.Each activity is a type of support project that is（　）by a particular problem,event, or opportunity encountered with the implemented system.

问题 1：	A.designed	B.implemented	C.investigated	D.analyzed
问题 2：	A.supported	B.tested	C.implemented	D.constructed
问题 3：	A.construction	B.maintenance	C.execution	D.implementation
问题 4：	A.broke	B.formed	C.composed	D.decomposed
问题 5：	A.triggered	B.leaded	C.caused	D.produced

4. You are developing a sever-side enterprise application. It must support a variety of different clients including desktop browsers , mobile browsers and native mobile applications. The application might also expose an API for 3rd parties to customer. It might also（1）with other applications via either web services or a message broker. The application handles requests（HTTP requests and messages）by executing business logic; accessing a database；exchanging messages with other systems; and returning a HTML /JSON/XML （2） . There are logical components corresponding to different functional areas of the application.
What's the application's deployment architecture?
Define an architecture that structures the application as a set of （3 ）, collaborating services. This approach corresponds to the Y-axis of the Scale Cube. Each service is:
Flighty maintainable and testable——enables rapid and frequent development and deployment.
Loosely coupled with other services——enables a team to work independently（the majority of time on their services）without being imported by changes to other services and without affecting other services.
（4 ） deployable——enable a team to deploy their services without having to coordinate with other teams.
Capable of being developed by a small team-essential for high productivity by avoiding the high communication head of large teams.
Services （5） using either synchronous protocols such as HTTP/REST or asynchronous protocols such as AMQP. Services can be developed and deployed independently of one another. Each service has its own database in order to be decoupled from other services. Data consistency between services is maintained using some particular pattern.

问题 1：A.integrate	B.Coordinate	C.cooperate	D.Communicate
问题 2：A.request	B.response	C.text	D.File
问题 3：A.loosely coupled	B.loosely cohesion	C.High coupled	D.Highly cohesion
问题 4：A.Dependently	B.Independently	C.Coordinately	D.Integratedly
问题 5：A.interoprate	B.coordinate	C.communicate	D.depend

5. A project is a [temporary]（ ）of unique, complex, and connected activities having one goal or purpose and that must be completed by a specific time, within budget, and according to（ ）.
Project management is the process of scoping, planning, staffing，organizing, directing, and controlling the development of a(n)（ ）system at a minimum cost within a specified time frame.
For any systems development project, effective project management is necessary to ensure that the project meets the（ ）, is developed within an acceptable budget, and fulfills customer expectations and specifications. Project management is a process that starts at the beginning of a project, extends through a project, and doesn't culminate until the project is completed.
The prerequisite for good project management is a well-defined system development process. Process management is an ongoing activity that documents, manages the use of, and improves an organization's chosen methodology (the "process")for system development. Process management is concerned with the activities, deliverables, and quality standards to be applied to（ ）project(s).

问题 1：	A.task	B.work	C.sequence	D.activity
问题 2：	A.specifications	B.rules	C.estimates	D.designs
问题 3：	A.perfect	B.acceptable	C.controlled	D.completed
问题 4：	A.deadline	B.specification	C.expectation	D.requirement
问题 5：	A.a single	B.a particular	C.some	D.all

6. The project workbook is not so much a separate document as it is a structure imposed on the documents that the project will be producing anyway.

All the documents of the project need to be part of this （1）. This includes objectives ,external specifications , interface specifications , technical standards , internal specifications and administrative memoranda（备忘录）.

Technical prose is almost immortal. If one examines the genealogy（手册）of a customer manual for a piece of hardware or software , one can trace not only the ideas , but also many of the very sentences and paragraphs back to the first （2） proposing the product or explaining the first design. For the technical writer, the paste-pot is as mighty as the pen.

Since this is so, and since tomorrow's product-quality manuals will grow from today’s memos, it is very important to get the structure of the documentation right. The early design of the project （3） ensures that the documentation structure itself is crafted, not haphazard. Moreover, the establishment of a structure molds later writing into segments that fit into that structure.

The second reason for the project workbook is control of the distribution of （4）. The problem is not to restrict information, but to ensure that relevant information gets to all the people who need it.

The first step is to number all memoranda, so that ordered lists of titles are available and each worker can see if he has what he wants. The organization of the workbook goes well beyond this to establish a tree-structure of memoranda. The （5） allows distribution lists to be maintained by subtree, if that is desirable.

问题 1：	A.structure	B.specification	C.standard	D.objective
问题 2：	A.objective	B.memoranda	C.standard	D.specification
问题 3：	A.title	B.list	C.workbook	D.quality
问题 4：	A.product	B.manual	C.document	D.information
问题 5：	A.list	B.document	C.tree-structure	D.number

7.Creating a clear map of where the project is going is an important first step. It lets you identify risks, clarify objectives, and determine if the project even makes sense. The only thing more important than the Release Plan is not to take it too seriously. Release planning is creating a game plan for your Web project （1） what you think you want your Web site to be. The plan is a guide for the content, design elements, and functionality of a Web site to be released to the public, to partners, or internally. It also （2） how long the project will take and how much it will cost. What the plan is not is a functional （3） that defines the project in detail or that produces a budget you can take to the bank.

Basically you use a Release Plan to do an initial sanity check of the project's （4） and worthiness. Release Plans are useful road maps, but don't think of them as guides to the interstate road system. Instead, think of them as the （5） used by early explorers—half rumor and guess and half hope and expectation.

It's always a good idea to have a map of where a project is headed.

问题 1：	A.constructing	B.designing	C.implementing	D.outlining
问题 2：	A.defines	B.calculates	C.estimates	D.knows
问题 3：	A.specification	B.structure	C.requirement	D.implementation
问题 4：	A.correctness	B.modifiability	C.feasibility	D.traceability
问题 5：	A.navigators	B.maps	C.guidances	D.goals

8.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（1）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（2）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（3）positive outcomes.

Once a（4）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（5）(as envisioned today, of course).

问题 1：	A.conflicts	B.consensus	C.success	D.disagreement
问题 2：	A.competition	B.agreement	C.cooperation	D.collaboration
问题 3：	A.total	B.complete	C.partial	D.entire
问题 4：	A.technology	B.standard	C.pattern	D.model
问题 5：	A.area	B.goal	C.object	D.extend

9.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（1 ）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（2）.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（3 ）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（4）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（5） .No! By doing things fast and quick,they make messes that make the software stiff,and hard to understand,beautiful systems are 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.

问题 1：A.Simple B.Hard C.Complex D.duplicated
问题 2：A.happens B.exists C.stops D.starts
问题 3：A.starts B.continues C.appears D.stops
问题 4：A.practical B.useful C.beautiful D.ugly
问题 5：A.impractical B.perfect C.time-wasting D.practical

10.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（1）, 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 （2）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（3）fashion,and the complexity of the whole increases much more than linearly.

The complexity of software is a(an)（4）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（5）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.

问题 1：	A.task	B.job	C.subroutine	D.program
问题 2：	A.states	B.parts	C.conditions	D.expressions
问题 3：	A.linear	B.nonlinear	C.parallel	D.additive0
问题 4：	A.surface	B.outside	C.exterior	D.essential
问题 5：	A.fixed	B.included	C.ignored	D.stabilized

11.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（1）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（2）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（3）access. An auditing function monitors and keeps a record of user accesses to system resources. In practice, a number of（4）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（5）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.

问题 1：	A.cooperates	B.coordinates	C.connects	D.mediates
问题 2：	A.denied	B.permitted	C.prohibited	D.rejected
问题 3：	A.open	B.monitor	C.grant	D.seek
问题 4：	A.component	B.users	C.mechanisms	D.algorithms
问题 5：	A.remote	B.native	C.controlled	D.automated

12.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（1）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 requirements, it's often（2）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（3）requirement capture approaches, with their emphasis on declarative requirements and "shall" statements,completely fail to capture the（4）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（5）.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.

问题 1：	A.plenty	B.loose	C.extra	D.strict
问题 2：	A.impossible	B.possible	C.sensible	D.practical
问题 3：	A.modern	B.conventional	C.different	D.formal
问题 4：	A.statics	B.nature	C.dynamics	D.originals
问题 5：	A.misapplied	B.applied	C.used	D.powerful

13.Why Have Formal Documents?
Finally, writing the decisions down is essential. Only when one writes do the gaps appear and the（1）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（2）directon, his chief daily task will be communication, not decision-making,and his documents will immensely（3）this load.
Finally,a manager,s documents give him a data base and checklist. By reviewing them（4）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（5）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.

问题 1：A.inconsistencies	B.consistencies	C.steadiness	D.adaptability
问题 2：A.other	B.different	C.another	D.same
问题 3：A.extend	B.broaden	C.lighten	D.release
问题 4：A.periodically	B.occasionally	C.infrequently	D.rarely
问题 5：A.decides	B.encapsulates	C.realizes	D.recognizes

14.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（1）for one person to do most engineering jobs. Systems development is a team（2）,and the effectiveness of the team largely determines the（3）of the engineering.
Development teams often behave much like baseball or basketball teams. Even though they may have multiple specialties, allthe members work toward（4）.However,on systems maintenance and enhancement teams, the engineers often work relatively independently, much like wrestling and track teams.
A team is（5）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.

问题 1：A.convenient	B.existing	C.practical	D.real
问题 2：A.activity	B.job	C.process	D.application
问题 3：A.size	B.quality	C.scale	D.complexity

问题 4：
A.multiple objectives　B.different objectives
C.a single objectives　D.independent objetives

问题 5：A.relatively	B../	C.only	D.more than

15.Cloud computing is a phrase used to describe a variety of computing concepts that involve a large number of computers（1）through a real-time communication network such as the Internet. In science, cloud computing is a（2）for distributed computing over a network, and means the（3）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（4）for building the platform layer of the cloud. In turn, the platform layer is a foundation for implementing the（5）layer for SaaS applications.

问题 1：	A.connected	B.imlemented	C.optimized	D.Virtualized
问题 2：	A.replacement	B.switch	C.substitute	D.synonym(同义词)
问题 3：	A.ability	B.applroach	C.function	D.method
问题 4：	A.network	B.foundation	C.software	D.hardware
问题 5：	A.resource	B.service	C.application	D.software

16.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.

问题 1：	A.almost	B.often	C.usually	D.barely
问题 2：	A.alternative	B.need	C.possibility	D.solution
问题 3：	A.design	B.redesign	C.plan	D.build
问题 4：	A.throwaway	B.system	C.software	D.product
问题 5：	A.worth	B.value	C.cost	D.invaluable

17.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.

问题 1：A.starts B.proceeds C.stops D.speeds
问题 2：A.change B.proceeds C.smooth D.hide
问题 3：A.frequently B.usually C.commonly D.rarely
问题 4：A.work B.program C.communicate D.talk
问题 5：A.worthless B.valueless C.useless D.invaluable

18.Computers will become more advanced and they will also become easier to use. Improved speed recognition will make the operation of a computer easier. Virtual 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.

问题 1：A.interact B.interacting C.communicate D.using
问题 2：A.interfaces B.behavior C.similarities D.comparison
问题 3：A.software B.properties C.programs D.hardware
问题 4：A.empirical B.real C.practical D.theoretical
问题 5：A.developments B.advantages C.limitations D.improvements

19.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.

问题 1：	A.hardware	B.computers	C.services	D.software
问题 2：	A.computers	B.disks	C.machines	D.tools
问题 3：	A.horizontally	B.vertically	C.inclined	D.decreasingly
问题 4：	A.clients	B.middleware	C.servers	D.hard disk
问题 5：	A.entire	B.individual	C.general	D.separate

20.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.

问题 1：	A.importance	B.keys	C.roles	D.values
问题 2：	A.small-sized	B.moderately-sized	C.large-sized	D.huge-sized
问题 3：	A.part-time	B.casual	C.seldom	D.full-time
问题 4：	A.together	B.by themselves	C.separately	D.alone
问题 5：	A.tell	B.know	C.communicate	D.feedback

21.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.

问题 1：	A.customers	B.managers	C.users	D.administrators
问题 2：	A.activity	B.procedure	C.process	D.progress
问题 3：	A.parts	B.modules	C.software	D.a whole
问题 4：	A.parts	B.modules	C.software	D.a whole
问题 5：	A.modified	B.used	C.suited	D.tailored

22. 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!”

问题 1：	A.creators	B.innovators	C.appliers	D.inventors
问题 2：	A.technical	B.classical	C.social	D.societal
问题 3：	A.trivial	B.crucial	C.minor	D.insignificant
问题 4：	A.technology	B.sociology	C.physiology	D.astronomy
问题 5：	A.technology	B.sociology	C.physiology	D.astronomy

23.

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.But when 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 than chefs, nor than other engineering managers. But false（ ）to match the patron's desired date 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.

问题 1：	A.no	B.the same	C.other	D.lots of
问题 2：	A.testers	B.constructors	C.managers	D.architects
问题 3：	A.tasks	B.jobs	C.works	D.scheduling
问题 4：	A.easy	B.difficult	C.simple	D.painless
问题 5：	A.sharing	B.excluding	C.omitting	D.ignoring

24.Virtualization is an approach to IT that pools and shares（）so that utilization is optimized and supplies automatically meet demand.Traditional IT environments are often silos， where both technology and human（）。 are aligned around an application or business function.With a virtualized（），people， processes， and technology are focused on meeting service levels， （）is allocated dynamically， resources are optimized， and the entire infrastructure is simplified and flexible.We offer a broad spectrum of virtualization（）that allows customers to choose the most appropriate path and optimization focus for their IT infrastructure resources.

问题 1：	A.advantages	B.resources	C.benefits	D.precedents
问题 2：	A.profits	B.costs	C.resources	D.powers
问题 3：	A.system	B.infrastructure	C.hardware	D.link
问题 4：	A.content	B.position	C.power	D.capacity
问题 5：	A.solutions	B.networks	C.interfaces	D.Connections

25.NAC’s（Network Access Control）role is to restrict network access to only compliant endpoints and（）users.However， NAC is not a complete LAN（）solution; additional proactive and（）security measures must be implemented.Nevis is the first and only comprehensive LAN security solution that combines deep security processing of every packet at 100Gbps，ensuring a high level of security plus application availability and performance.Nevis integrates NAC as the first line of LAN security（）In addition to NAC，enterprises need to implement role-based network access control as well as critical proactive security measures-real-time， multilevel （）inspection and microsecond threat containment.

问题 1：	A.automated	B.distinguished	C.authenticated	D.destructed
问题 2：	A.crisis	B.security	C.favorable	D.excellent
问题 3：	A.constructive	B.reductive	C.reactive	D.productive
问题 4：	A.defense	B.intrusion	C.inbreak	D.protection
问题 5：	A.port	B.connection	C.threat	D.Insurance

26.WebSQL is a SQL-like（）language for extracting information from the web.Its capabilities for performing navigation of web （）make it a useful tool for automating several web-related tasks that require the systematic processing of either ail the links in a （）， all the pages that can be reached from a given URL through （）that match a ’ pattern，or a combination of both.WebSQL also provides transparent access to index servers that can be queried via the Common （）Interface.

问题 1：	A.query	B.transaction	C.communication	D.programming
问题 2：	A.browsers	B.servers	C.hypertexts	D.clients
问题 3：	A.hypertext	B.page	C.protocol	D.operation
问题 4：	A.paths	B.chips	C.tools	D.directories
问题 5：	A.Router	B.Device	C.Computer	D.Gateway

27.Originally introduced by Netscape Communications， （）are a general mechanism which HTTP Server side applications， such as CGI （）， can use to both store and retrieve information on the HTTP （）side of the connection.Basically， Cookies can be used to compensate for the （） nature of HTTP.The addition of a simple， persistent， client-side state significantly extends the capabilities of WWW-based （）.

问题 1：	A.Browsers	B.Cookies	C.Connections	D.Scripts
问题 2：	A.graphics	B.processes	C.scripts	D.texts
问题 3：	A.Client	B.Editor	C.Creator	D.Server
问题 4：	A.fixed	B.flexible	C.stable	D.stateless
问题 5：	A.programs	B.applications	C.frameworks	D.Constrains

28.

For nearly ten years， the Unified Modeling Language (UML) has been the industry standard for visualizing， specifying， constructing， and documenting the（ ） of a software-intensive system.As the（ ）standard modeling language， the UML facilitates communication and reduces confusion among project （ ）.The recent standardization of UML 2.0 has further extended the language’s scope and viability.Its inherent expressiveness allows users to （ ） everything from enterprise information systems and distributed Web-based applications to real-time embedded systems.

The UML is not limited to modeling software.In fact， it is expressive enough to model（ ）systems， such as workflow in the legal system， the structure and behavior of a patient healthcare system， software engineering in aircraft combat systems， and the design of hardware.

To understand the UML， you need to form a conceptual model of the language， and this requires learning three major elements: the UML’s basic building blocks， the rules that dictate how those building blocks may be put together， and some common mechanisms that apply throughout the UML.

问题 1：A.classes	B.components	C.sequences	D.artifacts
问题 2：A.real	B.legal	C.de facto	D.illegal
问题 3：A.investors	B.developers	C.designers	D.stakeholders
问题 4：A.model	B.code	C.test	D.modify
问题 5：A.non-hardware	B.non-software	C.hardware	D.software

29.Why is（　）fun? What delights may its practitiopect as his reward? First is the sheer joy of making things. As the child delights in his mud pie， so the adult enjoys building things， especially things of his own design. Second is the pleasure of making things that are useful to other people. Third is the fascination of fashioning complex puzzle-like objects of interlocking moving parts and watching them work in subtle cycles， playing out the consequences of principles built in from the beginning. Fourth is the joy of always learning， which springs from the（　）nature of the task. In one way or another the problem is ever new， and its solver learns something:sometimes（　）， sometimes theoretical， and sometimes both. Finally， there is the delight of working in such a tractable medium. The（　）， like the poet， works only slightly removed from pure thought-stuff. Few media of creation are so flexible， so easy to polish and rework， so readily capable of realizing grand conceptual structures.

Yet the program（　）， unlike the poet’s words， is real in the sense that it moves and works， producing visible outputs separate from the construct itself. It prints results， draws pictures， produces sounds， moves arms. Programming then is fun because it gratifies creative longings built deep within us and delights sensibilities we have in common with all men.

问题 1：A.programming	B.composing	C.working	D.writing
问题 2：A.repeating	B.basic	C.non-repeating	D.advance
问题 3：A.semantic	B.practical	C.lexical	D.syntactical
问题 4：A.poet	B.architect	C.doctor	D.programmer
问题 5：A.construct	B.code	C.size	D.scale