First edition (early access)

Frequently Asked Technical Interview Questions and Answers Guide

www.acetheinterview.com

PUBLISHED BY:

JEEVE TECHNOLOGIES, LLC.
EMAIL: WEBMASTER@ACETHEINTERVIEW.COM
URL: WWW.ACETHEINTERVIEW.COM

TABLE OF CONTENT

Thank you for purchasing the most frequently asked technical interview questions and sample answers guide. The guide contains more than 100 great interview questions and their sample answers. Most of the questions listed in the guide are frequently asked by respected technology companies including Microsoft. In such a competitive marketplace where not answering one question can make a difference between success and failure, it is our hope that this guide will help prepare you for your next big interview and will help you get the job you deserve. We have tried to keep the document short and to the point so that you don't have to waste valuable time reading fluff.

As Interviews can be very dynamic and situation dependent, we would love to hear if this guide helped you better prepare for your interview. In addition, if you have any suggestions or ideas on how we can make it better, we would very much appreciate your feedback. Please write us at webmaster@acetheinterview.com or visit us at www.acetheinterview.com.

Good luck!

Part-I: Frequently Asked Interview Questions

Questions in an Information Technology related job interview can be organized into the following categories:

1. General
2. Fundamental
3. Analytical
4. Algorithms & Coding
5. Language/Technology specifics

The questions are more or less designed around the specific position you are interested in; however, you should expect some common questions irrespective of the job description. As in any typical interview, the questions below are not arranged in any particular order of difficulty. Please try to answer these questions on your own before looking at the sample answers in Part-II, as it will go a long way in getting you ready for your interview. In addition, keep in mind that this is not meant to be an exhaustive list of Interview questions- just some of the most frequently asked questions. You are encouraged to consult other sources as part of your interview preparation.

1. GENERAL

1) How would you describe yourself? What are your strengths and weaknesses? (Answer)
2) Tell me about some of the major projects you have worked on. What challenges did you encounter and how did you deal with them? (Answer)
3) What were your biggest (professional) mistakes and successes? What did you learn from them?
4) What is the biggest misconception about you amongst your colleagues? Or friends? (Answer)
5) What is your ideal job? (Answer)
6) Why are you looking to change jobs? Why did you decide to seek a position with this company? (Answer)
7) In what ways do you think you can make a contribution to our company?
8) How much salary are you expecting?
9) Where do you see yourself in the future, say 5 years from now? (Answer)
10) How well do you work in a team environment? Can you give me an example of when you disagreed with a team, and how you dealt with it? Describe a situation in which you had to lead a team of people over which you had no formal authority.

2. PROGRAMMING FUNDAMENTALS

1) How can computer technology be integrated into an elevator system for a hundred story office building? How do you optimize for availability? How would variation of traffic over a typical work week, floor, or time of day affect this? How would you test this system? (Answer)
2) Suppose we wanted to control different parts of a home using a computer. Describe the high level architecture of the system you would use to do this? (Answer)
3) List some of the problems unique to distributed databases? (Answer)
4) You are implementing the software for an IP router. When a packet comes in, the router has to decide which link to route the packet out on, based on its destination IP address. How would you implement this? How would you optimize for speed? How would you optimize for space? (Answer)
5) Find the fastest, simplest algorithm you can for determining if two rectangles overlap. (Answer)
6) What is the difference between multitasking and multithreading? (Answer)
7) How would you go about learning a new programming language under high-pressure conditions? (Answer)
8) What is disk interleaving? Why do you need it? (Answer)
9) What is a linked list? (Answer)
10) How will you test a vending machine? (Answer)
11) Explain Inheritance, Multiple Inheritance, Abstraction, and Polymorphism? (Answer)
12) How do you multiply an integer variable by 16 without using the multiplication, addition or division operator? What if you have to multiply by 15? (Answer)
13) What are the steps you take to write a program? (Answer)
14) What do people mean by event driven programming? Is the event-driven paradigm suitable for OOP?
15) What is a class factory? What are the advantages and disadvantages of a class factory?
16) Create a "String" class and show how to improve the copy constructor and assignment operator to improve efficiency.
17) What are some of the differences between a system call and a library call? (Answer)

3. ANALYTICAL

1) Why is a manhole cover round? (Answer)

2) If given a rectangular cake with a rectangular piece removed (any size or orientation), how would you cut the remainder of the cake into two equal halves with one straight cut of a knife? (Answer)

3) There are four people who need to cross a bridge at night. The bridge is only wide enough for two people to cross at once. There is only one flashlight for the entire group. When two people cross, they must cross at the slower member's speed. All four people must cross the bridge in 17 minutes, since the bridge will collapse in exactly that amount of time. Here are the times each member takes to cross the bridge:

   Person A: 1 minute
   Person B: 2 minutes
   Person C: 5 minutes
   Person D: 10 minutes

   If Person A and C crossed the bridge initially, 5 minutes would elapse, because Person C takes 5 minutes to cross. Then Person A would have to come back to the other side of the bridge, taking another minute, or six minutes total. Now, if Person A and D crossed the bridge next, it would take them 10 minutes, totaling to 16 minutes. If A came back, it would take yet another minute, totally 17. The bridge would collapse with A and B on the wrong side. How can all four people get across the bridge within 17 minutes? Note: there is no trick-answer to this problem. You cannot do tricky stuff like throwing the flashlight etc. (Answer)

4) You have someone working for you for seven days and you have one gold bar to pay them. The gold bar is segmented into seven connected pieces. You must give them a piece of gold at the end of every day. If you are only allowed to make two breaks in the gold bar, how do you pay your worker? (Answer)

5) You have 5 jars of pills. Each pill weighs 10 grams, except for contaminated pills contained in one jar, where each pill weighs 9 grams. Given a scale, how could you tell which jar had the contaminated pills in just one measurement? (Answer)

6) You have 12 balls. All of them are identical except one, which is either heavier or lighter than the rest. That odd ball is either hollow while the rest are solid or solid while the rest are hollow. You have a simple two-armed scale, and are permitted three measurements. Can you identify the odd ball and determine whether it is hollow or solid? (Answer)

7) This puzzle was apparently written by Einstein in the last century[1]. He said that 98% of the people in the world cannot solve the quiz. See if you can...

Facts:

1: There are 5 houses in 5 different colors

2: In each house lives a person with a different nationality.

3: These 5 owners drink a certain beverage, smoke a certain brand of cigar and keep a certain pet.

4: No owner has the same pet, smoke the same brand of cigar or drink the same drink.

Hints:

1: The British lives in a red house.

2: The Swede keeps dogs as pets

3: The Dane drinks tea

4: The green house is on the left of the white house (it also means they are next door to each other)

5: The green house owner drinks coffee

6: The person who smokes Pall Mall rears birds

7: The owner of the yellow house smokes Dunhill

8: The man living in the house right in the center drinks milk

9: The Norwegian lives in the first house

10: The man who smokes Blend lives next to the one who keeps cats

11: The man who keeps horses lives next to the man who smokes Dunhill

12: The owner who smokes Blue Master drinks beer

13: The German smokes Prince

14: The Norwegian lives next to the blue house

15: The man who smokes Blend has a neighbor who drinks water.

The question is: who keeps the fish? (Answer)

8) There are three wise men in a room: A, B and C. You decide to give them a challenge. Suspecting that the thing they care about most is money, you give them $100 and tell them they are to divide this money observing the following rule: they are to discuss offers and counter-offers from each other and then take a vote. The majority vote wins. Sounds easy enough... now the question is, assuming each person is motivated to take the largest amount possible, what will the outcome be? (Answer)

9) There are 2 rooms. The first has 3 switches (on/off) and the other one has 3 bulbs (electric lamps). Initially all the switches are off and all bulbs are turned off. Every bulb is connected to one switch and only one. First you are in the room with the switches. You can play with the switches however you want, but you can only enter the rooms with the bulbs once. Find the correspondence between the switches and the bulbs (Answer)

10) There is a prisoner who has two doors in front of him. One leads to freedom and one leads to hell. He has no idea which one is which. There are 2 guards there. One guard always tells the truth and one always lies. The prisoner can ask each guard one question (the same question to each) that will lead you to your freedom. What question should the prisoner ask? (Answer)

11) You have two jars, 50 red marbles and 50 blue marbles. A jar will be picked at random, and then a marble will be picked from the jar. Placing all of the marbles in the jars, how can you maximize the chances of a red marble being picked? What are the exact odds of getting a red marble using your scheme? (Answer)

12) You are given two 60 minute long fuse ropes (i.e. the kind that you would find on the end of a bomb) and a lighter. The fuses do not necessarily burn at a fixed rate. For example, given an 8 foot rope, it may take 5 minutes for the first 4 feet of the fuse to burn, while the last 4 feet could take 55 minutes to burn (a much slower rate) (5+55=60 minutes). Using these two fuses and the lighter, how can you determine 45 minutes? HINT: You can use the lighter any number of times (Answer)

13) A cable, 16 meters in length, hangs between two pillars that are both 15 meters high. The ends of the cable are attached to the tops of the pillars. At its lowest point, the cable hangs 7 meters above the ground. Question: How far apart are the two pillars? (Answer)

1

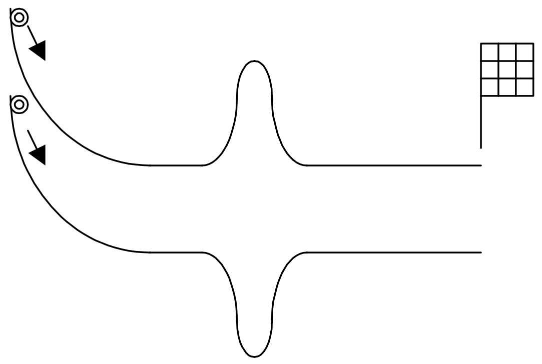
14) There was a sheriff in a town that caught three outlaws. He said he was going to give them all a chance to go free. All they had to do is figure out what color hat they were wearing. The sheriff had 5 hats, 3 black and 2 white. Each outlaw can see the color of the other outlaw's hats, but cannot see his own. The first outlaw guessed and was wrong so he was put in jail. The second outlaw also guessed and was also put in jail. Finally the third blind outlaw guessed and he guessed correctly. How did he know? (Answer)

15) One train leaves Los Angeles at 15 MPH heading for New York. Another train leaves from New York at 20mph heading for Los Angeles on the same track. If a bird, flying at 25mph, leaves from Los Angeles at the same time as the train and flies back and forth between the two trains until they collide, how far will the bird have traveled? (Answer)

16) Imagine that you have 26 constants, labeled A through Z. Each constant is assigned a value in the following way: A = 1; the rest of the values equal their position in the alphabet (B corresponds to the second position so it equals 2, C = 3, etc.) raised to the power of the preceding constant value. So, B = 2 ^ (A's value), or B = 2^1 = 2. C = 3^2 = 9. D = 4^9, etc., etc. Find the exact numerical value to the following equation:

(X - A) * (X - B) * (X - C) * ... * (X - Y) * (X - Z) (Answer)

17) Two balls begin rolling down the tracks below (see diagram). Which one reaches the end first, and which is moving faster at the end? Assume no friction or wind resistance. (Answer)



18) A farmer has $100.00. He wants 100 animals. What combination of the animals below will get the farmer EXACTLY 100 animals with EXACTLY $100.00? Cows are $2.00 each, Pigs are $0.50 each and Chickens are $0.10 each.

4. ALGORITHMS & CODING

1) You are developing a web browser (something like e.g. Netscape, etc.) and need to display all visited links on a page. The visited links need to use a different color then that used to display scheme than the unvisited links. Now, given a history of links you have visited before, how would you go about writing the piece of code that makes the determination if you have seen this link before? Answer or not? The answer could be a simple string comparison, but then think about the time it will take for the client to render any HTML page. Alternatively, so, given a history of URLs, come-up with an elegant way (algorithm, data structures, etc.) to make the determination if a given link already exists in the history list? (Answer)
2) Since web pages can have multiple URLs pointing to them, as a web browser developer how can you make sure you have never seen the same content before? (Answer)
3) Write a function to print all the possible permutations of a string. Now, modify the algorithm to discard duplicates. (Answer)
4) Design a memory management scheme.
5) Implement strstr(), strcpy(), strtok() etc. (Answer)
6) Write an algorithm and C code to find the sub array with the largest sum given an array that contains both positive and negative integers (Answer)
7) A square picture is cut into 16 squares and then shuffled. Write a program to rearrange the 16 squares to get the original big square. (Answer)
8) Implement an algorithm to reverse a singly linked list (with and without recursion). (Answer)
9) Count the total number of set bits in a number without using a loop. (Answer)
10) Given an array of characters which form a sentence of words, give an efficient algorithm to reverse the order of the words in it. (Answer)
11) Do the class/structure description for a Hash table, and write the source code for the insert function. (Answer)
12) What sort of technique you would use to update a set of files over a network, where a server contains the master copy.
13) How do you handle deadlock on a table that is fed with a live serial feed?

14) Say you have three integers between 0-9. You have the equation: A! +B! +C! = ABC (where ABC is a three digit number, not A*B*C). Find A, B, and C that satisfies this equation. (Answer)
15) Is it possible to keep 2 stacks in a single array if one grows from position one of the array, and the other grows from the last position? Write a procedure PUSH (x, s) that pushes element x onto stack S, where S is one or the other of these two stacks. Include all necessary error checks.
16) How would you find a cycle in a linked list? Try to do it in O(n) time. Try it using a constant amount of memory. (Answer)
17) Implement an algorithm to reverse a doubly linked list. (Answer)
18) The web can be modeled as a directed graph. Come up with a graph traversal algorithm. Make the algorithm non-recursive and breadth-first. (Answer)
19) How would you implement a queue from a stack? (Answer)
20) Write source code for printHex (int i) in C/C++. (Answer)
21) Given an array of size N in which every number is between 1 and N, determine if there are any duplicates in it. (Answer)
22) You are given one fixed list of numbers (let's call it the first list) and one other list (second list). How can you efficiently find out if there is any element in the second list that is an element of the first list? (Answer)
23) Give me an algorithm and C code to shuffle a deck of cards, given that the cards are stored in an array of ints. Try to come up with a solution that does not require any extra space. (Answer)
24) Do a breadth first traversal of a tree. (Answer)
25) Write a function to find the depth of a binary tree. (Answer)
26) Given 2 nodes in a tree, how do you find the common root of the nodes? (Answer)
27) Write atoi().

5. C/C++

1) Explain the mechanism of virtual functions and virtual function tables. (Answer)
2) Given a singly linked list and a pointer to a certain node in the list, how would you delete that node in constant time? (Answer)
3) What are recursive functions? What are the advantages and disadvantages of recursive algorithms? (Answer)
4) What leads to code-bloating in C++? (Answer)
5) What are references in C++? Why do you need them when you have pointers? (Answer)
6) How do you do dynamic memory allocation in C applications? List advantages and disadvantages of dynamic memory allocation vs. static memory allocation. (Answer)
7) What happens if an error occurs in a constructor or destructor? (Answer)
8) What are some of the main differences between a linked list and an array? (Answer)
9) Differentiate between a copy constructor and an assignment operator. (Answer)
10) What is multiple inheritance? What are the potential pitfalls of multiple inheritance? How would you avoid multiple inheritance? (Answer)
11) What are virtual destructors? (Answer)
12) What are constructors and destructors? (Answer)
13) What are virtual functions? (Answer)
14) What are the differences between struct, class and union? (Answer)
15) What is exception handling? What are the advantages of exception handling? (Answer)
16) What is the difference between new()/delete() and malloc()/free ()? (Answer)
17) Describe different types of polymorphism available in C++. (Answer)

6. JAVA

1) Describe what happens when an object is created in Java(Answer)
2) In Java, You can create a String object as below :

   String str = "abc"; & String str = new String ("abc");

   Why can't a button object be created as: Button bt = "abc". Why is it compulsory to create a button object as:

   Button bt = new Button ("abc")

   (Answer)
3) What is the advantage of OOP? (Answer)
4) What are the main differences between Java and C++? (Answer)
5) What are interfaces? (Answer)
6) How can you achieve Multiple Inheritance in Java? (Answer)
7) What is the difference between StringBuffer and String class? (Answer)
8) Describe, in general, how java's garbage collector works? (Answer)
9) What are some of the main differences between == operator and equals method? (Answer)
10) What are abstract classes and abstract methods? (Answer)
11) How can you force all derived classes to implement a method present in the base class? (Answer)
12) Java says "write once, run anywhere". What are some ways this isn't quite true? (Answer)
13) What is the difference between a Vector and an Array? Discuss the advantages and disadvantages of both? (Answer)
14) What does the keyword "synchronize" mean in Java? When do you use it? What are the disadvantages of synchronization? (Answer)
15) What is JDBC? Describe the steps needed to execute a SQL query using JDBC. (Answer)
16) Are constructors inherited? Can a subclass call the parent classe's constructor? When? (Answer)
17) Describe java's security model (Answer)

7. DATABASES

1) What is the difference between a "where" clause and a "having" clause? (Answer)
2) What is a "transaction"? Why are they necessary? (Answer)
3) What is "normalization"? "Denormalization"? Why do you sometimes want to denormalize? (Answer)
4) What types of join algorithms can you have? (Answer)
5) State some advantages and disadvantages of indexing database tables. (Answer)
6) Why can a "group by" or "order by" clause be expensive to process? (Answer)
7) What types of index data structures can you have? (Answer)
8) What are stored procedures? What are the advantages of stored procedures?

8. INTERNET TECHNOLOGY

1) You have two machines, both connected to the internet. One you are told has a web server running on it, the other doesn't have a web browser of any kind. Using the machine without the browser, how can you tell that the web server is running on the other machine? (Answer)
2) Explain how ping works. (Answer)
3) How does Ethernet work? How big is an Ethernet address? (Answer)
4) What is a bridge? A router? A gateway? (Answer)
5) What is TCP? UDP? How do they differ? (Answer)
6) What is XML and what is it good for? (Answer)
7) What is a "proxy server"? How does it work? (Answer)
8) What is a "cookie"? How do they work? What are the different types of Cookies? (Answer)
9) Describe the process that happens when you click a link in your web browser to fetch another web page. (Answer)

Part-II: Sample Answer Guide

Before answering any question, ask yourself what it is the interviewer is looking for? For example: when an interviewer asks an analytical question, most likely the interviewer is trying to test your creative thinking and analytical skills, and is not as concerned about one specific (correct) answer. So, be inventive and think out-loud.

Here is a short list of things you should keep in mind while answering questions in an interview: For more information please visit www.acetheinterview.com.

- **Skills required to perform the job**: It is important to emphasize the skills which you feel the employer is seeking. Good Interviewer's usually test the skill level required to perform a job in indirect ways. So, try to identify what it is that the interviewer is going after.
- **Work Experience**: If possible relate questions to work you have done in the past
- **Verbal communication skills**: This includes the ability to listen effectively, verbalize thoughts clearly, and express yourself confidently.
- **General personality**: For many employers, how your personality fits in with the rest of the company is as important as your skills to perform the job! So tailor your personality to fit the culture of the company.

1. GENERAL

1) These types of questions are often best answered by focusing on 5 to 6 of your strong selling points. The answer should include your achievements, skills and talents that would enable you to succeed in the position that is being discussed. When asking this question, the interviewer is not looking for your history of school work and employment, but is giving you a chance to sell yourself in about 60 seconds. It is important to that you limit your answers to 60 seconds or so if possible, because if you go on talking about your past at length the interviewer might loose interest and may not remember the key points you were trying to communicate. If you have sufficiently researched the organization, you should be able to find out what skills the company values the most. Make sure that your strengths cover most of these skills. While talking about weakness, be positive; turn a weakness into strength. For example, you might say: "I often worry too much over my work. Sometimes I work late to make sure the job is done well". (Return to question)

2) When going to an interview, you should always be prepared to speak about difficult situations that you have encountered in the past and how you were able to effectively resolve them. It is good to remember at least three. And remember, the situation should always reflect an attribute that you think the employer/interviewer would admire. Some companies train in "Behavioral Interviewing". Its premise is that past behavior is a very good predictor of future behavior, and the interviewing process focuses on asking lot of specifics on your past history. For example: "Tell me about a time you had a disagreement with a coworker, and how you dealt with the situation." The best preparation for this and many other interview questions is to have two or three projects that had many challenges, and be prepared with details on how you overcame them. (Return to question)

4) This is a hint for answering this question. Remember that perception is most often 90% reality. Be careful with your answer. (Return to question)

5) If you know what motivates you, try to frame that in the context of the position you are applying for. If you do not want to give a specific answer, then try preparing a short organized statement you are comfortable with. For example: "My ideal job is the one that I can contribute my skills or knowledge on a daily basis and, in the process, add significant value to the company ".
(Return to question)

6) Few questions are more important than these, so it is important to answer them clearly and with enthusiasm. Show the interviewer your interest in the company by sharing what you learned about the job, the company and the industry through your own research. Talk about how your professional skills will benefit the company. Unless you work in sales, your answer should never be simply "money"!! ☺ (Return to question)

9) The interviewer wants to know if your plans and the company's goals are compatible. Let him know that you are ambitious enough to plan ahead. Talk about your desire to learn more and improve your performance with time. Be as specific as possible about how you will meet the goals you have set for yourself. Also, keep in mind that the interviewer is mainly interested in what you can do for the company, not what the company can do for you. It helps if your long-term goals help the company too, for example "I would like go into management eventually, so I see myself as a manager for a successful product." If you don't have any specific plans you could use a prepared statement like "Five years from now, I see myself being an integral part of the this companies culture, being rewarded and appreciated for putting my best efforts forward, and geared up to take up a more challenging role here!" (Return to question)

2. PROGRAMMING FUNDAMENTALS

1) Implement a model similar to thread scheduling models found in most multitasking OS. Some points to consider:

- Some floor may have higher priorities, like the executive's floor for instance. Calls from those floors would always be serviced first.
- The elevator should only stop at a floor for x number of seconds, to avoid monopolizing the elevator. This time would be decreased during busy hours and increased during slow hours.
- If, for some reason, a floor has an elevator call outstanding for some long period of time, that floor's priority gets boosted so that it's serviced immediately.
- Based on weight (or # of people or some other similar metric) the elevator car could know when it's full and ignore calls until there is room for at least 2 people. Stopping for just one person isn't worth it for a full elevator.
- When the elevator is idle, it can go to the floor that will need the elevator the most or that which has the highest priority. This will save on the wait time at that floor (e.g.: the ground floor).
- Since it is a 100 story bldg, it is highly likely that many elevators are servicing it. So there needs to be coordination between the elevators to service properly. The elevators could be programmed so that they only serve certain floors or floor ranges. This divide and conquer strategy can be optimized by reducing the number of floors an elevator serves based on floor location (e.g.: the higher the floor, the less number of floors an elevator serves).
- In front of the entrance door of each floor, there could be placed a BIG poster/caricature depicting the health benefits of climbing the stairs!! This could reduce the elevator traffic.

An alternative solution could be to optimize for availability using the Poisson statistical model. The arrival of the people can be modeled as a Poisson distributed random variable with a set of curves being collected over a period of time (one set of curves for every day of the week) and keep on optimizing these set of curves using linear regression so as to obtain a single curve that is the "best-fit" for that day.

The elevator would thus stand at the floor where there is a maximum probability of availability of people. A variation of traffic would mean that the set of points over which the curve fitting is done would change, and thus the computer system would adapt to that change. This is closer to hard disk operation: reading the track/sector that is closer to the head position. Then apply the rule to avoid starvation. (Return to question)

3) Among other things, the following points must be considered while designing a distributed database:

- Storing data (fragmentation/replication, horizontal/vertical table partitioning)
- Catalog management (naming, data independence)
- Query processing (cost-based optimization, semi-join)
- Updating data (synchronous/asynchronous)
- Concurrency (distributed deadlock)
- Recovery (2 or 3 phase commit)

(Return to question)

4) Assuming that destination IP addresses are stored in a hash table, you can to do a lookup based on the destination IP address of the incoming packet. However, the question is what kind of algorithm to use to generate the hash?

The two ends of the spectrum are using a hash table of size 2^32 (32 for the 32-bit IP address) in which case the time complexity would be O (1) but the space complexity will be very high (not practical). On the other end you can use a binary search lookup, with time complexity of O (log n). The right fit will depend on the time/space trade-off the particular application in question can make. See a fantastic paper on this topic at http://citeseer.nj.nec.com/crescenzi99ip.html.
(Return to question)

5) The way to solve this is to know the coordinates of the points of the diagonal and compare them.

- If the coordinates are the same, then the rectangles "exactly overlap”.
- If one set of rectangular coordinates lies within the range of the other rectangle, then the rectangles overlap though not completely.

I'll leave the code as an exercise for the reader ☺. (Return to question)

6) They sound similar, but are very different. Multitasking refers to the ability of the operating system to execute multiple programs simultaneously. The processing time is shared between the programs. In preemptive multitasking, the OS divides the time between processes. In cooperative multitasking, the processes control the CPU for as long as it needs and then releases it for the next process to take ownership of the CPU. For example, multitasking in the Windows OS enables the user to use MS Word and MS Excel at the same time. Windows 2000 is preemptive, but Windows3.1 is cooperative (that's why one misbehaved application freezes the whole system).

Multithreading refers to the ability of the operating system to execute multiple parts of a process simultaneously. The threads usually have access to the resources allocated by the main thread (process). Multithreading is usually used to do background processing without affecting the responsiveness of the program. For example, the user can browse the internet while the browser is downloading a file because the download is performed by a different thread.

In some instances a pseudo multithreading can be created using multitasking, but the overhead of creating a new process as opposed to creating a new thread can be prohibitive.
(Return to question)

7) I would try and relate it to a programming language I already know. For example, if I need to learn C# and I already know Java I will try and relate it to Java. (Return to question)

8) To speed up the read access of the disk, the sectors are arranged such that when the computer finishes reading the sector, the disk head is on the position of the next sector to be read. Interleaving, with disk-based storage, refers to the physical arrangement of data sectors on a computer disk in such a way that sequentially read sectors are not necessarily contiguous. A disk, especially a hard disk, usually spins so fast that the computer cannot process the data from one sector before the next sector passes the head. Interleaving alternates sectors in a pattern that increases the likelihood that when the computer is ready for the next sector in numeric sequence, it will be the sequence just arriving at the head. For example, rather than being arranged in numeric order (1, 2, 3, 4 . . .) on the disk in a 1-to-1 interleave (no intervening sectors), sectors might be arranged in a 3-to-1 interleave pattern (1, 12,7,2,13,8,3 . . .), in which consecutive sectors are separated by two others. Interleaving speeds access by reducing the average time the computer must wait for the desired sector to arrive at the head. Interleaved sectors are arranged by the format utility that initializes a disk for use with a given computer. (Return to question)

9) A link list is a data structure which overcomes the problem associated with arrays (i.e. fixed size). The size of a link list can be manipulated dynamically. Where conventional lists arrange information in a linear sequence, linked lists are able to manage discontinuous bits of information through the use of pointers that link them together. This allows users (or list managers) to insert new pieces of information at any point in the list without having to disturb any of the existing pieces by merely adjusting pointers to link to the new value. (Return to question)

10) To test any software system, first you need the software requirements specification. So if you are asked to test a system, ask for the software requirements specification first. If there is no SRS, then you cannot test anything because you do not know what constitutes correct behavior.

Now for the Vending machine. After getting the SRS, take each requirement and expand it into test cases.

- When a quarter is inserted, the user is asked to choose an item.
- When an item is chosen, the item will fall through slot.
- If the item is unavailable, the user will be told so and asked for another choice.
- If the user hits cancel, the money is refunded.
- Correct change will be made if the user puts in more than the purchase price.
- Additional change will be requested if the user puts in less than purchase price.

Then proceed to test these requirements one at a time, using multiple choices and money inputs.

The following questions may also be asked in conjunction with the original question:

- How about if a person kicks the vending machine?
- What if there is a power outage??
- What if the machine is taken to Saudi Arabia? What changes should be done?

As the vending machine can be simulated with a finite state machine, with the states representing different status of the machine (e.g. how much money has been inserted), we can potentially automated testing by employing model-based testing technique (if this task needs to be performed repetitively). (Return to question)

11) Inheritance is deriving a class from another class. It is a mechanism for software reuse wherein you create specialized classes using base classes that already exist.

For example:
```
class a{
}
class b : public a {
}
```
So class 'a' is the base class and class 'b' is the derived class. (Return to question)

12) This is usually done by using bit-shifting operators (<< or >>). When you left shift a variable by 1 bit, you are essentially multiplying it by 2. Similarly, when you right shift a variable by 1 bit you are dividing it by 2. So, to multiply a variable by say 16, you can just left shift the variable by 4 bits (2*2*2*2 = 16). To multiply a variable by 15, you can multiply it by 16 (as above) and then subtract the original. Divide is very similar, just reversed. This operation may generate overflows/underflows, etc. A follow-up question can be how would you propose we handle these situations?

You might get extra points afterward by engaging a discussion on why using bit-shifts these days is not usually a good idea. At the hardware level, bit-shifts and adds/subtracts are much faster than multiplication, but modern optimizing compilers are smart enough to figure out the fastest way to perform mathematic operations. "x * 15" in your code is a lot clearer and less prone to bugs than "x << 4 - x", and modern compilers will generate similar code. (Return to question)

13) Refer to Code Complete or other similar titles by "Steve McConnell" (or any other book you know of) for a good Software Design Life-Cycle process. In short, the following steps generally make for a good SDLC:

- Start with clear requirement specification.
- Start designing with the core functionality in mind. The approach (top-down or bottom up) depends on whether the project is about writing a system from scratch or adding new functionalities.
- Implement the core functionality.
- Unit test core functionality to see if it is scalable and robust. Ensure that it has well defined interfaces for other modules.
- Start adding layers of other functionalities. Unit test and integrate the layers with the core functionality
- Keep working on the core functionality so as to make it more efficient.
- Many other aspects are also involved, but this should form the basis of good software.

(Return to question)

17) The primary difference between a system call and a library call is that library calls are standard across platforms whereas system calls are operating system dependent. In addition, library calls get linked with the application, whereas system calls are entry points to the operating system. (Return to question)

3. ANALYTICAL

1) There are many possible answers to this question. The interviewer is trying to test your creative thinking and analytical skills, so be inventive. Some answers are:

- The Man hole cover is round so it will not fall into the hole. If it were square it would be possible for the cover to fall into the hole if it was tilted at the proper angle. Being round there is no way, without breaking the cover, for it to fall into the hole. Round covers are more easily transported by one person because they can be rolled on their edge.
- The first person who designed it chose it to be round. Since he was the "first", it became a standard. People are more likely to follow standards rather than question them.
- The circular shape prevents the stress concentration at the edges, as circle does not have any edges.
- Round manhole will expand (heat & cold) evenly and not crack the surrounding concrete.
- Round holes are easier to bore.

(Return to question)

2) Two possible answers are:

- Find the centers of both the original cake and the removed piece. Cut the cake along the line connecting these two centers. As this line cuts the original cake and the removed piece in half, the remainder is two equal halves.
- Cut the entire cake in half horizontally (i.e. parallel to the table). This will get you two even halves.

(Return to question)

3) Two possible solutions are:

1. A and B cross together. Total Time: 2 Minutes
2. A comes back. Total Time: 3 Minutes.

3. C and D cross together. Total Time: 13 Minutes

4. B comes back. Total Time: 15 Minutes

5. A and B cross together. Total Time: 17 Minutes

Or:

1. A and B cross together. Total Time: 2 Minutes

2. B comes back. Total Time: 4 Minutes

3. C and D cross together. Total Time: 14 Minutes

4. A comes back. Total Time: 15 Minutes

5. A and B cross together. Total Time: 17 Minutes

As you can see from the solutions above, Key is for C and D to walk together.

(Return to question)

4) As 7 = 1 + 2 + 4, this implies there are only 2 divisions (cuts).

Day 1: Give [1] and you have [2+4]

Day 2: Give [2] and take back [1]. You have [1+4]

Day 3: Give [1+2] and you have [4]

Day 4: Give [4] and take back [1+2]. You have [1+2].

Day 5: Give [1+4] and you have [2]

Day 6: Give [2+4] and take back [1]. You have [1]

Day 7: Give [1+2+4] and you have nothing!

The same trick can be applied to:

15 Days = 1 + 2 + 4 + 8

31 Days = 1 + 2 + 4 + 8 + 16

And so on.

The main point of division is:

1 = 2^0

2 = 2^1

4 = 2^2

8 = 2^3

16= 2^4

And so on...

(Return to question)

5) Follow the steps outlined below:

Step-1: Mark the jars with numbers 1, 2, 3, 4, and 5.

Step-2: Take 1 pill from jar 1, take 2 pills from jar 2, take 3 pills from jar 3, take 4 pills from jar 4 and take 5 pills from jar 5.

Step-3: Put all of the jars on the scale at once and take the measurement.

Step-4: Now, subtract the measurement from 150 (1*10 + 2*10 + 3*10 + 4*10 + 5*10)

Step-5: The result will give you the jar number which has the contaminated pills.

(Return to question)

6) First number the balls 1, 2, 3 etc. Each can be L (light) or H (heavy). You need a paper to write down the inferences of each weighing.

Weighing 1

On the left side of the scale put balls 1, 2, 3, and 4

On the right side of the scale put balls 5, 6, 7 and 8

If the left side weighs less, than 1L or 2L or 3L or 4 L or 5H or 6H or 7H or 8H

(Remember the inference from this first step-it is used below)!

Different solutions for weighing 2 and 3 are given below depending on what happens in the second weighing.

NOTE: The same procedure may be followed if the right side weighs less in weighing 1 (balls 5, 6, 7 and 8).

Option 1

<u>Weighing 2</u>

On the left side of the scale put balls 1, 2, and 7

On the right side of the scale put balls 5, 4, and 9

If both sides are equal, then one of balls 3, 6 or 8 is bad.

<u>Weighing 3</u>

On the left side of the scale put balls 3 and 6

On the right side of the scale put balls 4 and 5

If both sides are equal then 8H is the solution

If the left side weighs less, then 3L or 6L. 6 cannot be both H and L, so 3L is the answer.

If the right side weighs less, then 3H or 6H. 3 cannot be both H and L, so 6H is the answer.

-OR –

Option 2

<u>Weighing 2</u>

On the left side of the scale put balls 1, 2, and 7

On the right side of the scale put balls 5, 4, and 9

If the left side weighs less, then 1L or 2L or 7L or 5H or 4H (9 is good)

<u>Weighing 3</u>

On the left side of the scale put balls 1and 5

On the right side of the scale put balls 4 and 3

If both sides are equal then balls 2 or 7 is bad. 2L is the answer, as 7 cannot be both H and L.

If the left side weighs less, then 1L or 5L. 1L is the answer.

If the right side weighs less, then 1H or 5H. 1 cannot be both H and L, so 5H is the answer.

-- OR --

Option 3

Weighing 2

On the left side of the scale put balls 1, 2, and 7

On the right side of the scale put balls 5, 4, and 9

If the left side weighs more, then 1H or 2H or 7H or 5L or 4L. Hence 7H or 4L is the answer, as 1, 2 and 5 cannot be both H and L.

Weighing 3

On the left side of the scale put ball 4

On the right side of the scale put ball 10

If both sides are equal, then 7H is the answer

If the left side weighs less, then 4L is the answer.

If the right side weighs less, as we have established that 4 is light and 10 is good.

(Return to question)

7) The fish is in the fourth house. Its green, the owner drinks coffee, smokes prince and he's German!

For those who want to know the approach for solving such problems:

First figure out is the order of the houses. We can use the following facts:

- The green house is on the left of the white house
- Blue is the second house.
- The center house is important because we know anything about it

So the possibilities could be:

____, Blue, ____, Green, White or ____, Blue, Green, White,____

There are 2 blanks and two colors. Red can't be the first house because the owner of the first house should be from Norway, and the owner of the red house is British.

That leaves us with:

1) Yellow, Blue, ____, Green, White – OR- 2) Yellow, Blue, Green, White, ____

The blank spot is the red house.

The 2nd option can't be true, as we have clues that the center house owner drinks milk, but the green house owner drinks coffee.

So we get the order as: Yellow, Blue, Red, Green, and White. Then make the table as follows

House/Attributes	Yellow	Blue	Red	Green	White
Drink					
Pet					
Smoke					
Nationality					

Once you have the right order then it's only the matter of reading the clues and filling in the table entries.

(Return to question)

8) It is unlikely that one wise man would be voted out because counter-offers are allowed. Consider: A and B decide to leave C out and split halfway. C offers B to leave A out instead, and as incentive offers B $60 as compared to $50 (Note: for C, $40 is better than nothing at all).

Now A will try to raise that figure, but keep no less than $33 for himself. It would also happen that as B starts to get more and more of the share, A and C would decide to keep B out instead and split amongst each other. This would go on and on until they reach equilibrium with each agreeing to take their rightful share of 33 dollars. (Return to question)

9) Turn one light switch (switch1) and leave it on long enough for the bulb to get warm. Then turn that switch off and turn another one on (switch2). Then go into the room with the bulbs. The light that is currently on belongs to switch2. One of the bulbs that is off will be warm since you left one of the switches on for awhile. That bulb belongs to switch1. The third bulb that is off belongs to the last switch that wasn't touched, or switch3. (Return to question)

10) The important thing here is that you know that one guard will tell the prisoner the truth and the other will lie, but the prisoner doesn't know who is who. So which guard should the prisoner ask? It doesn't matter; the prisoner just has to include both guards in the question. For example, the prisoner should ask "What would the other guard say the door to freedom is?" This way the prisoner will know the answer he is going to get will be wrong, and he can choose the other door. (Return to question)

11) Follow these steps:

- Put 1 red marble in the first jar.
- Put the other 49 reds and the 50 blues in the other jar (99 total marbles).
- The probability of selecting the jar with only 1 red marble and getting the only marble in the jar is 50% (1/2 * 1).
- The probability of selecting the other jar is 50%, and the odds of getting a red marble from that jar is 49/99 and (1/2 * 49/99). That equals 24.7475%.
- The total probability is 74.7475%.

(Return to question)

12) Light the first fuse at both ends and the other fuse at one end simultaneously. When the first fuse is burned out, you know that exactly half an hour has passes. You also know that the second fuse still has exactly half an hour to go before it will be burned completely, but we won't wait for that. Now also light the other end of the second fuse. This means that the second fuse will be burned completely after another quarter of an hour, which adds up to exactly 45 minutes since we started lighting the first fuse. (Return to question)

13) The height of the lowest point of the rope is 7m. The height of the pillars is 15m.

This determines the vertical displacement of the rope from the horizontal connecting the tops of the pillars calculate: 15 - 7 = 8m.

But the total length of the rope = 16m.

Therefore, the distance between the two pillars = 0m

(Return to question)

14) Lets look at it this way. Here are our possibilities:

1) BBB
2) BBW
3) BWB
4) WBB
5) WWB
6) BWW
7) WBW

Now we can eliminate # 6 because in this case the first outlaw would be sure to know that he had on a black hat. # 7 can be eliminated for the same reason for the second outlaw's guess. In # 2, the first outlaw has to see at least 1 black hat (if he saw two while hats he wouldn't have guessed wrong). From this we know that outlaw 2 or outlaw 3 has a black hat (possibly both). Now outlaw 2 has the same dilemma, but he knows that one or both of outlaw 2 and 3 has a black hat. He can see that outlaw 3 has a white hat so in that case he would guess black and be correct, but he didn't (since we know he guessed wrong). Given this, we can remove option 2 from consideration.

Options 1,3,4,5 all have outlaw 3 wearing a black hat. Thus, assuming that convicts 1 and 2 are as logical as possible, the only options left all have outlaw 3 wearing a black hat.

(Return to question)

15) The distance traveled by the bird can be calculated as:

1. Let's say the distance between LA and NY is d miles.
2. The time before which the trains will collide: d / (15+20) hours.
3. The distance traveled by the bird in that time: (d / 35) * 25 = 5*d/7 miles.

Assumptions made: The bird must follow the line of the track, remain at the same altitude, and the speed must be relative to the ground and not air speed.

(Return to question)

16) (X - A) * (X - B) * (X - C) * ... * (X - Y) * (X - Z) equals 0. This is because (X - X) is zero, and any integer multiplied by zero is zero. (Return to question)

17) The ball on the lower track gets there first. Both balls are traveling at the same speed at the end of the track. Hint: The ball on the lower track travels the same distance but at a higher average speed). (Return to question)

4. ALGORITHMS & CODING

1) Using a Hash Table is probably the most efficient way to do this. You can use several hashing algorithms. For example, checksum of a link can be used as a key for hashing. This will ensure o (1) order provided a good checksum algorithm is used. Whenever a page loads, we can parse all URL's from the page, take their checksum, and compare them with the hash table. Whichever links match are then displayed in a different color. (Return to question)

2) Make a list (or a binary tree) of hashes using MD5, SHA1 or a similar hash/digest algorithm of the pages you have visited. Then compare the digest of the current page to the hashes in the tree. A hash table is good here too! A hash table of other longer hashes is a quick, easy, and efficient solution. (Return to question)

3) Below is a sample implementation that prints all possible permutations of a string:

```
#include <iostream.h>
#include <string.h>

const int MAX_STR = 20;

void CopyStr(char *s2, char *s1, int i)
{
    for (int j = 0, k = 0, len = strlen(s2); j < len; j++) {
        if (i != j) {
            s1[k++] = s2[j];
        }
    }
    s1[k] = '\0';
}

void Permute(char *str1, char *str2)
{
    int len = strlen(str1);

    if (len == 1) {
       //If you do not want to print duplicates, put the result in a
```

```
            // hash and then print the hash table.
             cout << str2 << str1 << "\n";

             return;
        }

        for (int i = 0; i < len; i++) {

             char currentChar = str1[i];
             char str3[MAX_STR];
             CopyStr(str1, str3, i);

             int len2 = strlen(str2);
             str2[len2] = currentChar;
             str2[len2+1] = '\0';

             Permute(str3, str2);
             str2[len2] = '\0';
        }
    }

    int main ()
    {
        char someStr[MAX_STR], someOtherStr[MAX_STR];
        cout << "Enter a string(less then << MAX_STR << “characters:” ";
        cin >> someStr;
        someOtherStr[0] = '\0';
        cout << "\n\nThe permutations of " << someStr <<" are: \n";
        Permute(someStr, someOtherStr);
        return 1;
    }
```

There are other much more interesting and optimal ways to avoid printing duplicates. For example, if the For loop kept a "currentChar history" as it went through the loop, it could skip the call to Permute() if the character was already in its history, since it knows only duplicate permutations would be generated. This is both faster and much more memory-efficient, since the total number of permutations you'd have to store in the hash table grows exponentially with the length of the string – try running this on even "abcdefghij" and you'll get 10! = 3628800 permutations. At 10 bytes per word, that is 36 MB of RAM, plus the hash table overhead. A 15-letter word would take over 13 terabytes! As you can see, a hash table answer is obvious, but clearly is not practical.

(Return to question)

5) To see samples of this code you can look at the C Runtime code that ships along with VC++. The code is very efficient and well written. (Return to question)

6) The information to be kept while going through the array only once (O(n)) is the best sub array ending in the current position and best sub array found for the entire array until the current step. If necessary, the starting and ending position for the sub array can be kept. After the processing of the last element, the algorithm will have determined the sub array having the largest sum. The algorithm works fine even if there is no negative number in the array.

Algorithm: "arr" is the array of n integer. The function will return the largest sum and the limits of the sub array producing that value.

```
int GetBestSubArray(int* arr , int n , int* nBegin , int* nEnd)
{
    //starting/ending position of the best sub array for entire array
   int nBestBegin=0,nBestEnd=0;

   //starting/ending position of the best sub array ending in
   //current position

   int nCrtBegin=0,nCrtEnd=0;
   // index to loop in the array

   int nCrtIndex = 0;
   //the sum of whole best sub array and the sum of current best
   //sub array
   int nBestSum=0, nCrtSum=0;
   //Nothing to analyze, return invalid array indexes
   if (n == 0){
         (*nEnd)=(*nBegin)=-1;
         return 0;
   }
   nBestSum=nCrtSum = arr[nCrtIndex];

   for(nCrtIndex=1;nCrtIndex<n;nCrtIndex++) {
         //Compute the current largest sum
         if (nCrtSum<0) {
           nCrtSum = arr[nCrtIndex];
           nCrtEnd = nCrtBegin=nCrtIndex;
```

```
        }
        else {
          nCrtSum+= arr[nCrtIndex];
          nCrtEnd=nCrtIndex;
        }
        if (nCrtSum > nBestSum) {
             nBestSum = nCrtSum;
             nBestBegin=nCrtBegin;
             nBestEnd=nCrtEnd;
        }
   }
   *nBegin=nBestBegin;
   *nEnd=nBestEnd;
   return nBestSum;
}
//Sample driver
int main ()
{
     int arr[5] = { 1, 5, 6, -1, -2 };
     int nBegin =0;
     int nEnd = 0;
     int nBestSum =  GetBestSubArray(arr, 5, &nBegin, &nEnd);
     cout << "Sum=" << nBestSum << "Begin Index=" << nBegin
          << "End Index=" <<nEnd ;
     return 1;
}
```
(Return to question)

7) Suppose the original picture is 16 pieces and the individual pieces are numbered as follows.

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

The problem is then, given any one of these numbered pieces, how do you place each piece in the correct position in a two dimensional array.

Let the string str contain the shuffled pieces. Then -

```
int **restore(char const *str)
{
   int x,y,i;
   int final[4][4];
   for (i = 0, i < 16, i++) {
```

```
            if !(str[i]%4) {
              x = (int)(str[i]/4) - 1;
              y = 3;
            }
            else {
              x = (int)(str[i]/4);
              y = (int)(str[i]%4) - 1;
            }
            final[x][y] = str[i];
        }
        return final;
      }
```

Check:

```
say str[i] = 5
x = 1
y = 0
```

So final[1][0] = 5 which is the correct position in the actual picture

```
say str[i] = 12
x = 2
y = 3
So final[2][3] = 12
```

and so on.

(Return to question)

8) Below is a sample implementation that reverses a singly linked list using recursion.

The simplest way to do this with recursion is:

```
void rev(Node* head, Node* prev)
{
   if (!head) return;
   rev(head->next, head);
   head->next = prev;
}
```

Head is now pointing to the end instead of the start of the list. To fix that, you can use a wrapper function like below:

```
void reverse(Node* &p)
{
   if (!p) return;
   for (Node* tmp = p; tmp->next; tmp = tmp->next) { }
   rev(p, NULL);
   p = tmp;
}
```

An iterative algorithm to reverse a singly linked list is presented below:

```
struct node *revlist (struct node *root)
{
   struct node *np = root, *t, *r = 0;
   while (np) {
      root = np;
      t = np->next;
      np->next = r;
      r = np;
      np = t;
   }
   return root;
}
```

Note that an iterative solution to this problem is better than a recursive one, since every recursive call uses up more memory on the stack, and large lists can cause the stack to overflow. The iterative solution uses a constant amount of memory, so it'll work fine no matter how big the list is. It is also faster, because it avoids all of the recursive function calling overhead. In general, iterative solutions are usually better in practice than recursive solutions. (Return to question)

9) Sample implementation that counts total number of set bits using recursion is given below could be:

```
unsigned int number_of_ones(unsigned int n)
{
   if(n&1==1) {
      return 1+number_of_ones(n-1);
   }
   else {
      retrun number_of_ones(n/2);
   }
}
```

Using loops this can be accomplished as below (Note: The solution below doesn't loop over the whole 32 bits, usually, of an unsigned integer)

```
unsigned int bitcount(unsigned int x)
{
  unsigned int count;
  for(count = 0; x; x &= (x-1)) count++;
  return count;
}
```

You can also use a lookup table to count the number of set bits.

(Return to question)

10) One way to reverse the order of words is:

```
string word_reverse(char * data)
{
    char c = ' ';
    string s=data;
    int len=strlen(data);
    vector<string> output;
    int i=0;
    while(s[i]!='\0') {
        if(s[i]==c) {
            output.push_back(s.substr(0,i));
            s=s.substr(i+1,len);
            i=0;
            continue;
        }
        i++;
    }
    output.push_back(s);
    s.erase(0,s.length());
    string s1;
    for(int j=output.size()-1;j>0;j--) {
        s1+=output[j];
        s1+=" ";
    }
    s1+=output[0];
    return s1;
}
```

(Return to question)

11) A very simplistic and limited implementation of Hash table may look something like this:

For a 100 entry hash table of words, put in the arguments from the command line into the hash table:

```
#define MAX 100
```

```
char **hashTable;

int hashIndex(char *);

int main(int argc, char **argv)
{
   if(argc < 2) {
      printf("invalid input\n");
      exit(1);
   }
   hashTable = (char **)malloc(argc * sizeof(char *));

   for(i=0;i<argc;i++) {
      hashTable[i] = NULL;
   }

   while(count < argc) {
      index = hashIndex(argv[count]);
      if(!hashTable[index]) {
         hashTable[index] = (char *)malloc(strlen(argv[count])
                                        *sizeof(char));
         strcpy(hashTable[index], argv[count]);
      }
      else {
         //resolve collision
         count++;
      }
   }
   return 0;
}

int hashIndex(char *wd)
{
   int count, index=0;
   int len = strlen(wd);
   while(count < len) {
      index += atoi(wd[count]);
   }
   index = (int) index % MAX;
   return index;
}
```

(Return to question)

14) 145 = 1! + 4! + 5! (Return to question)

16) There are quite a few ways to solving this problem. Here are a couple of them:

If you have access to the code, add a new field "visited" to the node structure and initialize it with 0.

```
typedef struct {
    int visited;
    ...Other fields..
}
```

Now, start traversing through the linked-list checking for the value of this visited field. If the value is set to 0, set it to 1 and go to the next node in the list. If you find a node with its visited value set to 1 before you reach the end of the list, there is a cycle in the linked-list. If not, there is no cycle present in the list. As you can see this takes O (n) time but requires variable amount of memory.

To accomplish this task using constant memory is much more challenging at first. If you like simple yet challenging problems, try it yourself before looking at the answer below.

```
bool cycle_finder(Node *head)
{
    Node *p1, *p2;
    bool hasCycle = false;
    p1 = p2 = head;
    while (p2 && p2->next) {
        p1 = p1->next;
        p2 = p2->next->next;
        if (p1 == p2) {
            hasCycle = true;
            break;
        }
    }
    return hasCycle;
}
```

Note that p2 is moving through the list twice as fast as p1. If the cycle exists in the list, p1 and p2 will eventually be equal. Otherwise, the loop will terminate as soon as p2 reaches the end of the list. (Return to question)

17) Here is sample code to reverse a doubly linked list: using both, iterative and recursive, method is given below:

```
node* rev_dlist(node* n)
{
   while (n) {
      node* t = n->next;
      n->next = n->prev;
      n->prev = t;
      if (!t) break;
      n = t;
   }
   return n;
}
```

Using recursion, the above can be implemented as:

```
node* rev_dlist(node* n)
{
   if (n) {
      node* t = n->next;
      n->next = n->prev;
      n->prev = t;
      if (!t) return n;
      return rev_dlist(t);
   }
   return 0;
}
```

(Return to question)

18) Pseudo code to model the web as a graph may look something like this:

```
URLlist <- Queue for storing URLs.
URLlist.push(intial list of URLs)
while (!URLlist.empty())
{
   URL = URL.pop()
   if (seen(URL)) continue;
   setseen(URL)
   fetch URL
   for all the URL's in the page pop them on the URLlist
}
```

(Return to question)

19) Here is an algorithm to implement queue with two stacks:

```
stack stk1, stk2;

void push(element e)
{
    push e in stk1;
}

element pop()
{
    if(!stk2.empty()) {
        return stk2[top_stk2--];
    } else {
        while(!stk1.empty())
        {
            stk2.push(stk1.pop());
        }
        return stk2[top_stk2];
    }
}
```

(Return to question)

20) Printing hexadecimal value of a given integer using recursion can be accomplished as below:

```
printHex(unsigned int i)
{
    if(i ==0) return;
    printHex(i/16);
    if((i%16) < 10) {
        putchar('0' + i%16);
    }
    else {
        putchar('A' + i%16 - 10);
    }
}
```

(Return to question)

21) There are multiple ways to solve this problem. Here are a couple of them:

1. Sort the array and compare 1-1 elements.
2. Maintain a bit set with size N with the array[i] as the key, if a bit is already set then a duplicate is found.

(Return to question)

22) Hash each of the numbers in the fixed list to a hash table using a function that will return a distinct hash value for each of the numbers in this fixed list (size of the hash table itself is irrelevant, as long as it is larger than the number of numbers to be put in it).
In the second list, for each value, determine the hash value. Compare this to the number in that location in the hash table. If the number is the same, the element is in both lists. If the number in that location is different or does not exist, then the element in the second list is not in the first list.

(Return to question)

23) A sample implementation to shuffle a deck of cards might look like this:

```
void shuffle( int * cards)
{
    int curr, victim, size, temp;
    curr = 0;
    size = 52;
    // assuming it's a regular 52 card deck
    while(curr < size)
    {
        victim = rand() % size;
        cout << rand() << "\n";
        temp = cards[curr];
        cards[curr] = cards[victim];
        cards[victim] = temp;
        curr++;
    }
}
```

(Return to question)

24) Use a queue to traverse the tree, dequeueing a node when it is visited, and enqueueing its children. The pseudo code is given below:

```
traverseLevelOrder (Tree t) {
    Queue q = new Queue()
    q.enqueue(t)
    while !q.isEmpty() {
        Node current=q.dequeue()
        if (current<>NULL) {
            q.enqueue(current.left)
            q.enqueue(current.right)
        }
    }
}
```

(Return to question)

25) To find the depth of a binary tree, let us assume that each node of the tree is defined as:

```
struct Node
{
    int data;
    Node * left;
    Node * right;
};
```

Now, walk through the left and the right node using recursion to find the depth of the tree

```
int depth(Node * N)
{
    if (N == 0) return -1;
    int DL = depth(N->left);
    int DR = depth(N->right);
    return (DL > DR) ? DL+1:DR+1;
}
```

(Return to question)

26) Use depth first traversal if no parent pointer, else find the height of the 2 nodes, normalize and then walk upwards together. (Return to question)

5. C/C++

1) Whenever a class member function is declared as virtual, the compiler creates a virtual table in memory which contains all function pointers that are declared as virtual in that class. This enables run time polymorphism (i.e. finding out the desired function at run time). Virtual function tables also have an additional pointer in the object to the vtable. As this additional pointer and the vtable increases the size of the object, a class designer needs to be judicious about declaring functions virtual. The sequence of events upon calling a method on the base object pointer is:

- Get vtable pointer (this vtable pointer points to the beginning of the vtable).
- Get the function pointers in the vtable using offset.
- Invoke the function indirectly through the vtable pointer.

(Return to question)

2) There are two things that we require to delete a node: the previous node's address and the next node. In this case we only know the current nodes address. So to delete it without breaking the list: Assume that the node comprises of

Data = The data

Next = Pointer to the next node

Also, assume that our linked list looks like this and we only know the current node address.

PREVIOUS NODE -> CURRENT NODE -> NEXT NODE

```
//Now Copy the contents of the next node to the current node…that's it
nextnode=currentnode->next
currentnode->data = nextnode->data
currentnode->next = nextnode->next
```

This does not work if the current node is the last node in the list.

(Return to question)

3) A function that calls itself repeatedly, satisfying some condition, is called a Recursive Function. We can't actually declare that recursive or non-recursive algorithms are good or bad. Some problems inherently are better suited for recursion. Examples include Fibonacci series generation or factorial of a number. Some advantages of recursive algorithms are:

- They are smaller in size (in terms of source code)
- They tend to be more elegant

Some disadvantages of recursion are:

- Among other things, recursion requires more of stack than non-recursive algorithms (due to several activation stacks for each call of the function)
- Coming-up with the correct algorithm requires a lot of careful thinking
- Testing recursive functions can be quite challenging

(Return to question)

4) Inline functions and templates, if not used properly, may lead to code bloating. Multiple Inheritance may also lead to code bloating (this is because the sub classes will end up getting members from all the base classes even if only few members will suffice). Techniques to avoid code blot are discussed in "Effective C++ programming". (Return to question)

5) A reference variable is actually just a pointer that reduces syntactical clumsiness related with pointers in C (reference variables are internally implemented as a pointer; it's just that programmers can't use it the way they use pointers).

As a side note, a reference must refer to some object at all times, but a pointer can point to NULL. In this way, references can be more efficient when you know that you'll always have an object to point to, because you don't have to check against NULL:

```
void func(MyClass &obj) {
    obj.Foo();
}

Is better than:
void func(MyClass *obj) {
    if (obj) obj->Foo();
}
```

(Return to question)

6) In C, malloc, calloc and realloc are used to allocate memory dynamically. In C++, new(), is usually used to allocate objects. Some advantages and disadvantages of dynamic memory allocation are:

Advantages:

- Memory is allocated on an as-needed basis. This helps remove the inefficiencies inherent to static memory allocation (when the amount of memory needed is not known at compile time and one has to make a guess).

Disadvantages:

- Dynamic memory allocation is slower than static memory allocation. This is because dynamic memory allocation happens in the heap area.
- Dynamic memory needs to be carefully deleted after use. They are created in non-contiguous area of memory segment.
- Dynamic memory allocation causes contention between threads, so it degrades performance when it happens in a thread.
- Memory fragmentation.

(Return to question)

7) Constructors don't have a return type, so it's not possible to use error codes. The best way to signal constructor failure is therefore to throw an exception. However, keep in mind that the memory for the object itself is released, and the destructors for all sub-objects (i.e. members and base classes) whose constructors have successfully run to completion will be called. The destructor for the object being constructed will not be called.

For example (assume appropriate definitions):

```
class T
{
   U u;
   public:
   T() { throw "Bye bye"; }
};

{
    T * tptr=new T;
}
```

In this example, the memory allocated by operator new _will_ be released, and u's dtor will be called.

```
class T2
{
   U *uptr;
   public:
   T2() { uptr=new U; throw "Bye bye";}
   ~T2() { delete uptr; }
};
```

With this class, the memory occupied by T2 will be release, *BUT* T2's dtor will _not_ be called, and the U object pointed to by uptr will be leaked.

(Return to question)

8) The main differences between a linked list and an array are:

- Arrays are faster in access than a link list for random access with index.
- Arrays are not dynamic while a link list is.
- Arrays are easier to sort than a link list.
- The elements of link list can be deleted/inserted while arrays cannot.

- Arrays occupy the same block of memory, while a link list is distributed.
- Array objects are automatically created by a compiler, while link lists are not.
- Arrays are part of most compilers, while link lists are not.
- Arrays are syntactically simple to read.

(Return to question)

9) The copy constructor is used to copy an object to a newly created object. This is used during initialization and not during ordinary assignment. The copy constructor is invoked whenever a new object is created and initialized to an existing object of the same kind.
In other words, the assignment operator handles assigning one object to another of the same class. If a statement creates a new object it is using initialization. If it alters the value of an existing object it is assignment.
(Return to question)

10) Deriving a class from more than one direct base class is called multiple inheritance. In the following example, classes A, B, and C are direct base classes for the derived class X:

```
class A { /* ... */ };
class B { /* ... */ };
class C { /* ... */ };
class X : public A, private B, public C { /* ... */ };
```

The order of derivation is relevant only to determine the order of default initialization by constructors and cleanup by destructors.

Potential pitfalls of Multiple Inheritance are:

1. Ambiguity
2. It's slow
3. The "Common Ancestor" problem. For example, if class B and class C derived from class A and if class D derived from class B and class C, then class D will have 2 copies of

class A that might lead to inconsistency, as the class doesn't know which copy it is viewing.

```
A
/ \
B C
\ /
D
```

Now D has 2 copies of class A. (Return to question)

11) Destructor implemented by declaring a base class's destructor with the keyword virtual is called a virtual destructor. A virtual destructor ensures that, when delete is applied to a base class pointer or reference, it calls the destructor implemented in the derived class, if an implementation exists.
Let's take the simplest polymorphic relation: A - base class, B - class derived from A. If we've got a pointer (or reference) to class A, but under the hood it is an object of type B, and we're trying to delete the object, declaration of virtual destructor in class A ensures that the destructor of class B will be called.

```
B* b = new B;
A* a = b //due to polymorphism!
delete a; // both A and B destructors are called.
```

(Return to question)

12) Constructors and destructors are provisions for initialization and cleanup of objects.
A constructor is a special member function with the same name as the Class. It is invoked automatically when the object is created. It usually contains initialization code for member variables and allocation of memory. There can be multiple overloaded constructors, with different input arguments, used to initialize the object in a variety of ways.

A destructor is a special member function that is called just before an object is destroyed. For example, when the object variable goes out of scope. It is used to perform cleanup. There can be only one destructor. Its name is '~' followed by the class name. (Return to question)

13) Virtual functions are functions whose behavior is known at runtime rather than at compile time. Due to this behavior, it can be said that virtual functions implement Polymorphism. In other words, preceding a function name with virtual in the base class means that that function is intended to be re-implemented (overridden) in the sub-class. (Return to question)

14) Struct, class and union all contain data members and methods. However, a struct and union have their member's public by default, while the class members are private by default. Also, a struct cannot contain an instance of itself. A union cannot be used as a base class in inheritance. None of a union's data members can be declared static and none of its functions can be virtual.
(Return to question)

15) Exceptions are an alternative to function return values. The big differences are:

- Exceptions cannot be ignored. They must be caught or the app will crash. It is a way of forcing the caller of a function to deal with an exceptional condition.
- It is also an improvement over return values, because you can put all possible values of your return type to good use, instead of having to dedicate one or more values as the "invalid" value.
- In addition, exceptions allow you to jump out of deeply nested function calls conveniently, avoiding a lot of return type checking and conditional statements.

(Return to question)

16) The main difference is that malloc() and free() don't know anything about constructors and destructors, where as new and delete do. The following lists the main differences:

- new automatically computes the size of the data object. In malloc you would have to use the sizeof operator.
- new automatically returns the correct pointer type. In malloc you would have to use a type cast.
- with new you can initialize the object while creating the object.
- new and delete can be overloaded.
- It's safe to delete a NULL pointer, but you'll get core dump to free a NULL pointer.

(Return to question)

17) Polymorphism has two types in C++:

1. Compile time polymorphism
2. Runtime Polymorphism

Operator overloading and Function Overloading are the examples for compile time polymorphism. Using Virtual Functions we will achieve run time polymorphism. (Return to question)

6. JAVA

1) Several things happen in a particular order to ensure the object is constructed properly:

- Memory is allocated from heap to hold all instance variables and implementation-specific data of the object and its superclasses. Implementation-specific data includes pointers to class and method data.
- The instance variables of the objects are initialized to their default values.
- The constructor for the most derived class is invoked. The first thing a constructor does is call the constructor for its superclasses. This process continues until the constructor for java.lang.Object is called, as java.lang.Object is the base class for all objects in java.
- Before the body of the constructor is executed, all instance variable initializers and initialization blocks are executed. Then the body of the constructor is executed. Thus, the constructor for the base class completes first and constructor for the most derived class completes last. The methods of the class and its parent hierarchy are made available. Lastly, the address of the object is returned.

(Return to question)

2) String is not just another class in Java. There are a lot of special cases in String class. If you notice carefully, String is an Immutable class where as that's not true for Button or most of the other classes. Java compiler as well as '+' and '+=' operators convert quoted characters in to String. I believe Java designers wanted to treat Strings as close to primitive types as possible and hence some of the differences.

Important to not that you are NOT calling a java.lang.String constructor when you type String s = "abc"; for example:

```
String x = "abc";
String y = "abc";
```

refer to the same object.

While

```
String x1 = new String ("abc");
String x2 = new String ("abc");
```

refer to two different objects.

(Return to question)

3) You will get varying answers to this question depending on whom you ask. Major advantages of OOP are:

- simplicity: software objects model real world objects, so the complexity is reduced and the program structure is very clear;
- modularity: each object forms a separate entity whose internal workings are decoupled from other parts of the system;
- modifiability: it is easy to make minor changes in the data representation or the procedures in an OO program. Changes inside a class do not affect any other part of a program, since the only public interface that the external world has to a class is through the use of methods;
- extensibility: adding new features or responding to changing operating environments can be solved by introducing a few new objects and modifying some existing ones;
- maintainability: objects can be maintained separately, making locating and fixing problems easier;
- re-usability: objects can be reused in different programs.

(Return to question)

4) Main differences between C++ and Java are:

- Everything is an object in Java(Single root hierarchy as everything gets derived from java.lang.Object)
- Java does not have all the complicated aspects of C++ (For ex: Pointers, templates, unions, operator overloading, structures etc...)
- The Java language promoters initially said "No pointers!", but when many programmers questioned how you can work without pointers, the promoters began saying "Restricted

pointers." You can make up your mind whether it's really a pointer or not. In any event, there's no pointer arithmetic.

- There are no destructors in Java (automatic garbage collection).
- Java does not support conditional compile (#ifdef/#ifndef type).
- Thread support is built into java but not in C++.
- Java does not support default arguments.
- There's no scope resolution operator: in Java. Java uses the dot for everything, but can get away with it since you can define elements only within a class. Even the method definitions must always occur within a class, so there is no need for scope resolution there either.
- There's no "goto" statement in Java.
- Java doesn't provide multiple inheritance (MI), at least not in the same sense that C++ does.
- Exception handling in Java is different because there are no destructors.
- Java has method overloading, but no operator overloading. The String class does use the + and += operators to concatenate strings and String expressions use automatic type conversion, but that's a special built-in case.
- Java is interpreted for the most part and hence platform independent.

(Return to question)

5) Interfaces provide more sophisticated ways to organize and control the objects in your system. The interface keyword takes the abstract concept one step further. You could think of it as a "pure" abstract class. It allows the creator to establish the form for a class: method names, argument lists, and return types, but no method bodies. An interface can also contain fields. An interface says: "This is what all classes that implement this particular interface will look like." Thus, any code that uses a particular interface knows what methods might be called for that interface, and that's all. So the interface is used to establish a "protocol" between classes (some object-oriented programming languages have a keyword called protocol to do the same thing).

A typical example is listed below (from "Thinking in Java" by Bruce Eckels):

```
import java.util.*;
interface Instrument {
     int i = 5; // static & final
     // Cannot have method definitions:
     void play(); // Automatically public
     String what();
     void adjust();
}

class Wind implements Instrument {
     public void play() {
     System.out.println("Wind.play()");
     }
     public String what() { return "Wind"; }
     public void adjust() {}
}
```

(Return to question)

6) Java's interface mechanism can be used to implement multiple inheritance, with one important difference from c++ way of doing MI: the inherited interfaces must be abstract. This obviates the need to choose between different implementations, as with interfaces there are no implementations.

Example:

```
interface CanFight {
   void fight();
}

interface CanSwim {
   void swim();
}
interface CanFly {
   void fly();
}
class ActionCharacter {
   public void fight() {}
}
class Hero extends ActionCharacter implements CanFight, CanSwim,
CanFly {
   public void swim() {}
   public void fly() {}
}
```

You can even achieve a form of multiple inheritance where you can use the *functionality* of classes rather than just the interface:

```
interface A {
   void methodA();
}

class AImpl implements A {
   void methodA() { //do stuff }
}

interface B {
   void methodB();
}

class BImpl implements B {
   void methodB() { //do stuff }
}

class Multiple implements A, B {
   private A a = new AImpl();
   private B b = new BImpl();
   void methodA() { a.methodA(); }
   void methodB() { b.methodB(); }
}
```

This completely solves the traditional problems of multiple inheritance in C++ where name clashes occur between multiple base classes. The coder of the derived class will have to explicitly resolve any clashes. (Return to question)

7) A string buffer implements a mutable sequence of characters. A string buffer is like a String, but can be modified. At any point in time it contains some particular sequence of characters, but the length and content of the sequence can be changed through certain method calls.
The String class represents character strings. All string literals in Java programs, such as "abc" are constant and implemented as instances of this class; their values cannot be changed after they are created. (Return to question)

8) The Java runtime environment deletes objects when it determines that they are no longer being used. This process is known as garbage collection. The Java runtime environment supports a garbage collector that periodically frees the memory used by objects that are no longer needed. The Java garbage collector is a mark-sweep garbage collector that scans Java's dynamic memory areas for objects, marking those that are referenced. After all possible paths to objects are investigated, those objects that are not marked (i.e. are not referenced) are known to be garbage and are collected (A more complete description of our garbage collection algorithm might be "A compacting, mark-sweep collector with some conservative scanning").

The garbage collector runs synchronously when the system runs out of memory, or in response to a request from a Java program. Your Java program can ask the garbage collector to run at any time by calling System.gc(). The garbage collector requires about 20 milliseconds to complete its task so, your program should only run the garbage collector when there will be no performance impact and the program anticipates an idle period long enough for the garbage collector to finish its job. Note: Asking the garbage collection to run does not guarantee that your objects will be garbage collected. The Java garbage collector runs asynchronously when the system is idle on systems that allow the Java runtime to note when a thread has begun and to interrupt another thread (such as Windows 95). As soon as another thread becomes active, the garbage collector is asked to get to a consistent state and then terminate. (From "The Java Language Tutorial by Sun").
(Return to question)

9) The equals method can be considered to perform a deep comparison of the value of an object, whereas the == operator performs a shallow comparison. The equals method compares what an object points to rather than the pointer itself (if we can admit that Java has pointers). This indirection may appear clear to C++ programmers. Remember to override the toString() method provided by java.lang.Object when you want to use equals() method on custom objects as for an Object the toString method simply returns the memory address and comparing two memory address is not what you want when checking for Object equality. String class comes with an implementation

of toString () which does the character by character comparison, so you don't have to worry about toString () while using equals on string objects. (Return to question)

10) Simply speaking a class or a method qualified with "abstract" keyword is an abstract class or abstract method. You create an abstract class when you want to manipulate a set of classes through a common interface. All derived-class methods that match the signature of the base-class declaration will be called using the dynamic binding mechanism.

If you have an abstract class, objects of that class almost always have no meaning. That is, abstract class is meant to express only the interface and sometimes some default method implementations, and not a particular implementation, so creating an abstract class object makes no sense and are not allowed (compile will give you an error message if you try to create one).
An abstract method is an incomplete method. It has only a declaration and no method body. Here is the syntax for an abstract method declaration:

```
abstract void f();
```

If a class contains one or more abstract methods, the class must be qualified an abstract. (Otherwise, the compiler gives you an error message.). It's possible to create a class as abstract without including any abstract methods. This is useful when you've got a class in which it doesn't make sense to have any abstract methods, and yet you want to prevent any instances of that class.

Abstract classes and methods are created because they make the abstractness of a class explicit, and tell both the user and the compiler how it was intended to be used.
For example:

```
abstract class Instrument
{
   int i; // storage allocated for each
   public abstract void play();
   public String what()
   {
        return "Instrument";
   }
```

```
      public abstract void adjust();
   }
   class Wind extends Instrument
   {
       public void play()
       {
          System.out.println("Wind.play()");
       }
       public String what()
       {
          return "Wind";
       }
       public void adjust()
       {
       }
   }
```
(Return to question)

11) An abstract Class is the way to go. An Interface is absolutely the wrong way to go. The best way to think of an Interface is just as what it is, a completely abstract class. One main difference between an interface and an abstract class is that in an abstract class you can define some methods but not all (though you still can't instantiate an abstract class). In an interface you can only declare methods, but you cannot define them.

In sum, an abstract class is the way to go. This will allow you to leave the single method in question as an abstract class and allow it to be defined by a subclass. (Return to question)

12) There are some issues that require you to spend some effort porting a system written for one OS to run on another OS:

- You have the ability to use JNI, which will use native libraries, which can work differently on different systems.
- You have the ability to make system calls with Runtime.getRuntime ().exec ("<system command>") which can make the code hardware dependent, e.g. "ls" will not work on Windows.

- You have the ability to use XY coordinates in Pixels instead of specifying a Layout Manager in Swing, which will change the way the GUI looks on different systems.
- There are also some bugs in VM implementations on different systems which may cause your program to run well on Sun but crash on HP if your program runs into a bug on the HP VM that is not in the SUN implementation. I have personally run into this.
- You have to perform separate performance tuning, e.g. garbage collection parameters, initial heap size, etc. for each OS you run your system on, depending on the performance of each OS and the hardware it runs on.
- Threads are implemented differently on each system. For example, on Windows even low priority threads will get a few timeslices, while on UNIX a low priority thread will never run if high priority threads do not yield to it.
- There are other minor operating differences as well, e.g. you send a kill -S SIGQUIT command on UNIX to see a thread dump, and on Windows you press the Break key.

(Return to question)

13) Main differences between Vector and Arrays are:

- Vector can contain objects of different types whereas array can contain objects only of a single type.
- Vector can expand at run-time, while array length is fixed.
- Vector methods are synchronized while Array methods are not.

Use ArrayList if you want functionality similar to a vector, but do not need synchronization..

(Return to question)

14) Synchronize is used when u want to make your methods thread safe. The disadvantage of synchronize is it will end up in slowing down the program. Also if not handled properly it will end up in dead lock.

There are few more things one should know about synchronization:

- Only use (and minimize its use) synchronization when writing multithreaded code as there is a speed (up to five to six time slower, depending on the execution time of the synchronized/non-synchronized method) cost associated with its use.
- In case of synchronized method modifier, the byte code generated is the exact same as non-synchronized method. The only difference is that a flag called ACC_SYNCRONIZED property flag in method's method_info structure is set if the synchronized method modifier is present.
- Also, synchronized keyword can make the code larger in size if used in the body of the method as bytecode for monitorenter/monitorexit is generated in addition to any exception handling.

So, bottom line is use synchronization very carefully, understand the performance implications, and prefer synchronize method modifier to synchronized blocks. (Return to question)

15) JDBC is java based API for accessing data from the relational databases. JDBC provides a set of classes and interfaces for doing various database operations.
The steps are:

- Register/load the jdbc driver with the driver manager.
- Establish the connection thru DriverManager.getConnection();
- Fire a SQL thru conn.executeStatement()
- Fetch the results in a result set
- Process the results
- Close statement/result set and connection object.

(Return to question)

16) Constructors are not inherited. The first statement in the constructor of the sub-class should be the call to the constructor of the super class.

Ex:
```
class A
{
	int i;
	A(int a)
	{
	   i=a;
	}
}
class B extends A
{
	B()
	{
	   super(10);//This is must.
	}
}
```

The first statement in the constructor of the sub-class should be a call to the constructor of the super class, only and only if, the base-class constructor has arguments. However if your base-class constructor does not have arguments, then the base-class constructor is automatically called in the sub-class. (Return to question)

17) Java has a "sand-box" security model, where un-trusted code can be kept away from data that it should not be able to touch, using a fine-grained analysis of individual operations.
This is as opposed to a "trusted user" security model, which is what Microsoft has been pushing with ActiveX, etc. In this model, the emphasis is on obtaining a chain of certificates of trusted users. Once something is trusted, it has full access to do all operations.
This link has more details:
http://java.sun.com/docs/books/tutorial/security1.2/overview/index.html
(Return to question)

7. DATABASES

1) "Where" is a restriction statement. You use where clause to restrict data being accessed from the database. Where clause is used before result is retrieved. But having clause is used after retrieving the data. Having clause is a kind of filtering command. You should always use a WHERE clause in preference to a HAVING clause, if possible. (Return to question)

2) A transaction is a series of data manipulation operations mostly triggered from the client end that must be committed into the database as one whole operation. This means that a transaction is complete only when all the series of data manipulations involved have been successfully committed into the database. All transactions must follow the ACID rules.

- Atomic - All operations execute or No operations execute.
- Consistent - After the operations execute, database is in a consistent state as it was before execution.
- Isolated - Appropriate locks are placed on shared data in order to isolate the operation on the data.
- Durability - Data must be persisted on hard disk, so that if there is a crash the data is durable.

(Return to question)

3) Normalizing data means eliminating redundant information from a table and organizing the data so that future changes to the table are easier. Denormalization means allowing redundancy in a table. The main benefit of denormalization is improved performance with simplified data retrieval and manipulation. This is done by reduction in the number of joins needed for data processing. (Return to question)

4) There are three kinds of JOINS in SQL Server, Nested table, Merge and Hash joins. All of the above depend upon the resources available, MERGE and HASH consume lots of memory; so, a system that is low on memory would sparingly resort to the above two join methodologies. Nested table joins are based on the fact that indexes exist on nested tables in a SELECT query, for each row in the outer table, a scan is done for the rows available on the inner tables' rows. If a match is found, that row is retrieved. (Return to question)

5) Advantages: Faster querying of data
Disadvantages: Slower Insert, Slower Updates (if you are also updating the primary key). Apart from the problems mentioned above most indexing structures suffer from the "the dimensionality curse" which is the exponential growth of data volume as a function of dimensionality as a result of which query processing becomes much slower. (Return to question)

6)Processing of "group by" or "order by" clause often requires creation of Temporary tables to process the results of the query. Which depending of the result set can be very expensive. (Return to question)

7)An index helps to faster search values in tables. The three most commonly used index-types are:

- B-Tree: builds a tree of possible values with a list of row IDs that have the leaf value. Needs a lot of space and is the default index type for most databases.
- Bitmap: string of bits for each possible value of the column. Each bit string has one bit for each row. Needs only few space and is very fast.(however, domain of value cannot be large, e.g. SEX(m,f); degree(BS,MS,PHD)
- Hash: A hashing algorithm is used to assign a set of characters to represent a text string such as a composite of keys or partial keys, and compresses the underlying data. Takes longer to build and is supported by relatively few databases.

(Return to question)

8. INTERNET TECHNOLOGY

1) The common mistake people make here is they answer "ping the machine". That doesn't check anything other than to see if the machine is visible on the network. It has nothing to do with the web server itself. To find out if the web server is running, just telnet to port 80. If the port 80 is open, it doesn't mean that it is the web server listening to that port yet. To make sure it is the web server, you can type a primitive HTTP request like “GET /” and press "enter" twice. If it is the web server, then you should get the index page HTML code on your screen. (Return to question)

2) The Packet Internet Groper (ping) utility on the host machine sends data-grams, or packets, addressed to the Internet Protocol (IP) address of the target host using the Internet Control Message Protocol (ICMP) echo request command. If the specified host is operational, it will respond using the ICMP echo reply. The ping utility will then display statistics about the number of lost packets and the amount of time it takes for the response to return from the target host. This utility only verifies that the bottom two layers, Network Interface Layer and Internet Layer, of the TCP/IP stack on the target machine are operational. (Return to question)

3) Ethernet is normally a shared media LAN. All stations on the segment share the total bandwidth, which is 10 Mbps (Ethernet), 100 Mbps, (Fast Ethernet) or 1000 Mbps (Gigabit Ethernet). With switched Ethernet, each sender and receiver pair has the full bandwidth.

Ethernet uses the CSMA/CD technology to broadcast each frame onto the physical medium (wire, fiber etc.). All stations attached to the Ethernet are "listening," and the station with the matching destination address accepts the frame and checks for errors. It is a data link protocol (MAC layer protocol) and functions at layers 1 and 2 of the OSI model. A unique number is assigned to each Ethernet network adapter. It is a 48-bit number maintained by the IEEE. (Return to question)

4) A bridge is an intermediate system used to connect two LANs that use similar LAN protocols. The bridge acts as an address filter, picking-up packets from one LAN that are intended for a destination on another LAN and passing those packets on. It operates at layer 2 of the OSI model.

A router used to connect two networks that may or may not be similar. Router employs an internet protocol present in each router and each end system of the network. It operates at layer 3 of the OSI model.

A gateway performs protocol conversion between different types of networks or applications. For example, a gateway can convert a TCP/IP packet to a NetWare IPX packet and vice versa, or from AppleTalk to DECnet, from SNA to AppleTalk and so on. Gateways function at layer 4 of the OSI model. (Return to question)

5) TCP is a connection oriented protocol. It means that a connection should remain between peer to peer entities in two systems for transfer of information. UDP is a connectionless protocol and sends the information in form of packets. Each packet can take different route.
(Return to question)

6) XML is Extensible Markup Language. XML was designed to describe data and to focus on what data is. It is important to know that XML was designed to store, carry and exchange data. It was not designed to display data. With XML, data can be exchanged between incompatible systems. With XML, plain text files can be used to store and share data. (Return to question)

7) Also called a "proxy" or "application level gateway," it is an application that breaks the connection between sender and receiver. All input is forwarded out a different port, closing a straight path between two networks and preventing a hacker from obtaining internal addresses and details of a private network. Proxy servers are available for common Internet services; for example, an HTTP proxy is used for Web access, and an SMTP proxy is used for e-mail. Proxies generally employ network address translation (NAT), which presents one organization-wide IP address to the Internet. It funnels all user requests to the Internet and fans responses back out to the appropriate users. Proxies may also cache Web pages, so that the next request can be obtained locally. Proxies are only one tool that can be used to build a firewall. (Return to question)

8) Cookies are small pieces of data used by web servers to help identify web users. There two types of cookies: Persistent and Non-Persistent.

- Persistent: Which remains on the user's machine even after user leaves the site. When they come back web server queries the cookie and if found identifies the user
- Non-Persistent: This cookie is used by web server to track the user information during a session. It expires once you leave the website

(Return to question)

9) The URL of the link is given to the browser engine, which opens a TCP/IP connection to the address listed in the URL. Assuming the URL specifies the HTTP protocol, and assuming no port is given in the URL, the connection is made to port 80 of the remote host. Various options may be specified in the HTTP headers which effect the expected behavior between client and host, an important and common option being "Connection: Keep-alive"; if this option is specified, the same connection is used to transfer all resources on the page (images, frames, sounds, flash animations, etc.), thus avoiding the overhead of establishing a separate TCP/IP connection for each resource the page requires to be loaded. Requests are made by issuing an HTTP "GET" command, followed by the resource requested. A single GET command may cause the client to request multiple resources (hence the existence of the keep-alive option). Finally, under normal circumstances, the connection

is closed after all resources are done loading ("Connection: Close" option is specified in the client's final request header). (Return to question)