1 :: What do you mean by analysis and design?

Analysis:
Basically, it is the process of determining what needs to be done before how it should be done. In order to accomplish this, the developer refers the existing systems and documents. So, simply it is an art of discovery.
Design:
It is the process of adopting/choosing the one among the many, which best accomplishes the users needs. So, simply, it is compromising mechanism.

2 :: What are the steps involved in designing?

Before getting into the design the designer should go through the SRS prepared by the System Analyst.
The main tasks of design are Architectural Design and Detailed Design.
In Architectural Design we find what are the main modules in the problem domain.
In Detailed Design we find what should be done within each module.

3 :: What are the main underlying concepts of object orientation?

Objects, messages, class, inheritance and polymorphism are the main concepts of object orientation.

4 :: What do u meant by SBI of an object?

SBI stands for State, Behavior and Identity. Since every object has the above three.
State:
It is just a value to the attribute of an object at a particular time.
Behaviour:
It describes the actions and their reactions of that object.
Identity:
An object has an identity that characterizes its own existence. The identity makes it possible to distinguish any object in an unambiguous way, and independently from its state.

5 :: Differentiate persistent & non-persistent objects?

Persistent refers to an object's ability to transcend time or space. A persistent object stores/saves its state in a permanent storage system with out losing the information represented by the object.
A non-persistent object is said to be transient or ephemeral. By default objects are considered as non-persistent.

6 :: What are models and meta models?

Model:
It is a complete description of something (i.e. system).
Meta model:
It describes the model elements, syntax and semantics of the notation that allows their manipulation.

7 :: What do you meant by static and dynamic modeling?

Static modeling is used to specify structure of the objects that exist in the problem domain. These are expressed using class, object and USECASE diagrams.
But Dynamic modeling refers representing the object interactions during runtime. It is represented by sequence, activity, collaboration and statechart diagrams.

8 :: How to represent the interaction between the modeling elements?

Model element is just a notation to represent (Graphically) the entities that exist in the problem domain. e.g. for modeling element is class notation, object notation etc.
Relationships are used to represent the interaction between the modeling elements.
The following are the Relationships.
Association: Its' just a semantic connection two classes.
e.g.:

Aggregation: Its' the relationship between two classes which are related in the fashion that master and slave. The master takes full rights than the slave. Since the slave works under the master. It is represented as line with diamond in the master area.

ex:
car contains wheels, etc.

car

Containment: This relationship is applied when the part contained with in the whole part, dies when the whole part dies.
It is represented as darked diamond at the whole part.
example:

```
class A{
//some code
};

class B
{
A aa; // an object of class A;
// some code for class B;
};
```

In the above example we see that an object of class A is instantiated with in the class B. so the object class A dies when the object class B dies.we can represnt it in diagram like

this.

Generalization: This relationship used when we want represents a class, which captures the common states of objects of different classes. It is represented as arrow line pointed at the class, which has captured the common states.
Dependency: It is the relationship between dependent and independent classes. Any change in the independent class will affect the states of t

9 :: Why generalization is very strong?

Even though Generalization satisfies Structural, Interface, Behaviour properties. It is mathematically very strong, as it is Antisymmetric and Transitive.
Antisymmetric: employee is a person, but not all persons are employees. Mathematically all As' are B, but all Bs' not A.
Transitive: A=>B, B=>c then A=>c.
A. Salesman.
B. Employee.
C. Person.
Note: All the other relationships satisfy all the properties like Structural properties, Interface properties, Behaviour properties.

10 :: Differentiate Aggregation and containment?

Aggregation is the relationship between the whole and a part. We can add/subtract some properties in the part (slave) side. It won't affect the whole part.
Best example is Car, which contains the wheels and some extra parts. Even though the parts are not there we can call it as car.
But, in the case of containment the whole part is affected when the part within that got affected. The human body is an apt example for this relationship. When the whole body dies the parts (heart etc) are died.

11 :: Can link and Association applied interchangeably?

No, You cannot apply the link and Association interchangeably. Since link is used represent the relationship between the two objects.
But Association is used represent the relationship between the two classes.
link :: student:Abhilash course:MCA
Association:: student course

12 :: Why is planning too much up front a mistake in an OOSAD?

You cant plan only for the current phase of the project as
your future activities are still coarse granular. To have
good plannig you need to have fine granularity w.r.t the
tasks to get clear WBS

13 :: Why should project managers complete hard problems first in an OOSAD project?

The query actually holds good in general for every situation in life. It is one of the principles of good time management.

The idea is to tackle hard (and important) problems first. This, if resolved - will pep up your confidence to deal with other not so hard issues. Also, this could have cascading effect on other issues that may get resolved on its own.

I would rather stress on "important" than "hard" issues. If a "hard" problem is not coming in the way of your deliverables (means it is not important) - keep it aside. There is no need to spend a lot of time on it.

14 :: Why does the function arguments are called as signatures?

The arguments distinguish functions with the same name (functional polymorphism). The name alone does not necessarily identify a unique function. However, the name and its arguments (signatures) will uniquely identify a function.
In real life we see suppose, in class there are two guys with same name, but they can be easily identified by their signatures. The same concept is applied here.
ex:

```
class person
{
public:
char getsex();
void setsex(char);
void setsex(int);
};
```

In the above example we see that there is a function setsex () with same name but with different signature.