Difference b/w Abstract class and Interface?
  Interface:: If we dont know anything about implementation just we have requirement specification then we should go for interface.
  Abstract class: If we are talking about implementation but not completely then we should go for abstract class.

  Interface:: Every method present inside the interface is always public and abstract whether we are declaring or not.
  Abstract :: Every method present inside abstract class need not be public and abstract.

  Interface:: We can't declare interface methods with the modifiers like private,protected,final,static,synchronized,native,strictfp.
  Abstract :: There are not restrictions on abstract class method modifiers.

  Interface:: Every interface variable is always public static final whether we are declaring or not.
  Abstract:: Every abstract class variable need not be public static final.

  Interface:: Every interface variable is always public static final we can't declare with the
                        following modifiers like
private,protected,transient,volatile.
  Abstract:: No restriction on access modifiers

  Interface:: For every interface variable compulsorily we should perform initialisation at the time of declaration,
            otherwise we get compile time error.
  Abstract:: Not required to perform initialisation for abstract class variables at the time of declaration.

  Interface:: Inside interface we can't write static and instance block.
  Abstract :: Inside abstract class we can write static and instance block.

  Interface:: Inside interface we can't write constructor.
  Abstract :: Inside abstract class we can write constructor.

Interview Questions
+++++++++++++++++++
Q>What is the difference b/w abstract class and interface?

Q>Every method present inside the interface is abstract,but in abstract class also we can take only abstract methods also then what is the need
   of interface concept?
Ans. we can replace interface with abstract class,but it is not a good programming practise.if we try to do, it would result in
     "missusing the role" of abstract class and it would also create peformance issue.

Using interface
+++++++++++++++
```
interface ISample
{

}
class SampleImpl implements ISample
{

}
```

1. ISample sample=new SampleImpl();//one level chaining :: SampleImpl ---> Object[Performance is relatively high]
2. While Implementing ISample, we can also get the benefit from Another class[Inheritance : Reusability].

Using Abstract class
+++++++++++++++++++

```
abstract Sample
{

}
class SampleImp extends Sample
{

}
```

1. Sample sample=new SampleImp();//Multi level chaining:: SampleImp ---> Sample ---> Object [Performance is low]
2. While extending Sample, we can't get the benefit of other classes[Inheritance can't be used here]

Q> Why abstract class can contains constructor and interface doesn't contains constructor?
 => Constructor :: To initialize the instance variable of an object, meaning is to provide values for instance variables.
            :: In abstract class we have instance variable so we need constructor for initializing the instance variables.
            :: In case of interface, we don't have instance variable we have variables which are of type
                  public static final, these variables are initialized at the time of declaration only.
               so we dont' need constructors in interface.

Q> When to go for interface and when to go for abstract class?
   interface -> To promote 100 percent abstraction we need to go for "interface" or to provide Software Requirement Specification we need to
            go for "interface".
   abstract class -> If we are taking about implementation that is partial implementation, then we need to go for "abstract class".

Which of the following are valid?
  1. The purpose of the constructor is to create the object.
  2. The purpose of the constructor is to initialize the object,not to create the object.
  3. Once constructor completes then only object creation completes.
  4. First object will be created and then constructor will be executed.
  5. The purpose of the new keyword is to create object and the purpose of constructor is to initalize the object.
  6. We can't create Object for abstract class directly but indirectly we can create.
  7. Whenever we are creating child class object automatically parent class object will be created.
  8. Whenever we are creating child class object automatically abstract class constructor(provided if it is parent) will be executed.
  9. Whenever we are creating child class object automatically parent constructor will be executed but parent object wont be created.
  10. Either directly or indirectly we can't create Object for abstract class and hence constructor concept is not applicable

for abstract class.

11. Interface can contain constructor.

Valid : 2,4,5,8,9

Invalid : 1,3,6,7,10,11