12 COMPUTER PROGRAMMING

INTERFACE and PACKAGE

Interface usage

```
public interface Sleeper {
  public long ONE_SECOND = 1000;
  public long ONE_MINUTE = 60000;
  public void wakeup();
 }
public interface Worker {
  public long WORK_TIME = 8;
  public void sleep();
}
public class Man implements Sleeper, Worker  {
  public void wakeup() {
    System.out.println("빨리 일어나 !!");
  }
  public void sleep() {
    ……
  }
}
```

Interface usage - ex1

```
interface IStack {
    void push(int item);
    int pop();
}
class FixedStack implements IStack {
    private int stack[];
    private int tos;
    FixedStack(int size) {
      stack = new int[size];
      tos = -1;
    }
    public void push(int item) {
      if(tos==stack.length-1)
          System.out.println("스택이 꽉찾음");
      else
          stack[++tos] = item;
     }
     public int pop() {
       if(tos < 0) {
          System.out.println("스택이 비었음");
          return 0;
       }
       else
          return stack[tos--];
     }
}
```

```
class InterfaceTest {
  public static void main(String args[]) {
    FixedStack mystack1 = new FixedStack(10);
    FixedStack mystack2 = new FixedStack(5);
    for(int i=0 ; i<10 ; i++)
        mystack1.push(i);
    for(int i=0 ; i<5 ; i++)
        mystack2.push(i);
    System.out.println("스택 : mystack1");
    for(int i=0 ; i<10 ; i++)
      System.out.print(mystack1.pop() + "  ");
    System.out.println();
    System.out.println("스택 : mystack2");
    for(int i=0 ; i<5 ; i++)
      System.out.print(mystack2.pop() + "  ");
  }
}
```

Interface Inheritance

```
public interface Sleeper {
    public long ONE_SECOND = 1000;
    public long ONE_MINUTE = 60000;
    public void wakeup();
}

public interface Worker {
    public long WORK_TIME = 8;
    public void sleep();
}

public interface People extends Sleeper, Worker {
    public int MAX = 24;
    public int MIN = 0;
    public void work();
}
```

Interface Reference - ex

```
interface A {
  int CONS = 5;
  public void display(String s);
}
class A1 implements A {
  int a = 10;
  public void display(String s) {
    System.out.println("display 메소드 구현 " + s);
  }
}
class InterTest {
  public static void main(String args[]) {
    A interfaceA;
    interfaceA = new A1();
    interfaceA.display("인터페이스 테스트");
    System.out.println("A의 상수 CONS의 값은 "+interfaceA.CONS);
    System.out.println("A1의 a 값 출력"+interfaceA.a);
}
}
```

Operator Instanceof – ex

```
class A {
   int i, j;
}
class B extends A{
   int k;
}
class C extends B {
   int l;
}

class InstanceOf {
  public static void main(String args[]) {
    A a = new A();
    B b = new B();
    C c = new C();
    if(a instanceof A)
       System.out.println("a는 A 클래스의 객체");
    if(b instanceof B)
       System.out.println("b는 B 클래스의 객체");
    if(c instanceof C)
       System.out.println("c는 C 클래스의 객체");
    if(c instanceof A)
       System.out.println("c는 A 클래스의 객체 : 형변환");
    if(a instanceof C)
       System.out.println("a는 C 클래스의 객체 : 형변환");
    else
       System.out.println("a는 C 클래스의 객체가 아님 : 형변환 불가");
  }
}
```