

## Section 18: Abstract Classes and Interfaces

### 1) Abstract classes contain methods that may be complete or abstract

```
public abstract class Aclass {  
    public abstract type method-name() {
```

- a) abstract classes must be inherited to be used
- b) abstract methods must be implemented by the user

### 2) Interfaces

- a) `public interface interface-name {`
- b) like a class, but with the bodies of the methods omitted and no variables but those in parameters
- c) must be implemented
- d) implementing class must implement methods using method headers

### 3) Interface Runnable

- a) interface Runnable
- b) to use, implement the interface Runnable

```
public final class class-name implements Runnable {
```

- c) imposes the methods `start()`, `run()`, `update()` and `stop()`
  - i) `start` is called first, used to start thread
  - ii) `run` is called next
  - iii) `update` is called when `repaint()` is called
  - iv) `stop` is called when `run()` finishes

Examples:

```
Thread animate;  
animate = new Thread();  
  
public void start() {  
    animate.start  
}  
  
public void stop() {;}  
  
public void update(Graphics g) {  
    paint(g);  
}
```

```
public void run() {  
    while (true) {  
        repaint();  
        pause(50);  
    }  
}
```