Stack		
0x100		0x200
[42]]	
T.T		
Heap		
0x600		0x900
]
Stack		
0x100		0x200
[0x600]
		_
Heap		
0x600		0x900

```
F() {
...F()...
G() {
F()
H() {
 F() ... F() ...
```

```
F() {
 if (...) {
  if (....) {
  if (...) {
```

```
Class A {
   f(B_IF b) { b.service1() }
interface B_IF {
  R1 service1(...);
  R2 \text{ service2}(...);
Class B1 implements B_IF {
```

Class B2 implements B_IF {

```
A.java:
Class A {
 f() {
    return 1;
B.java:
Class B {
 Aa;
 f () { a.f(); ... }
G(A a) \{ x = a.f(); .... \}
```

```
for(i in X) {
}
```

```
Interface I: a class with only function signatures
Class A implements I
Class A {
  f() code
  g() code
Class B extends A {
```

g() code'

Interface I: a class with only function signatures

Class A implements I

1,2,3

n+m

```
Public class ConsolePrinter implements Printer {
Public class ConsoleFactory() {
  Printer makePrinter() { return new ConsolePrinter(); }
Interface ConsoleFactoryIF {
 Printer makePrinter();
```

```
Class Worker {
    private Printer printer;
    Worker(Printer printer) { this.printer = printer; }
```

Two roles of Interface

- (1) Hiding the implementation details
- (2) Allowing generic programming

- (1) Program against Interface
- (2) Composition over Inheritance

```
r.setHeight(x);
r.setHeight(y);
assert(r.getArea() == x*y);
```

Inheritance: Abstraction/Polymorphism + Code Reuse

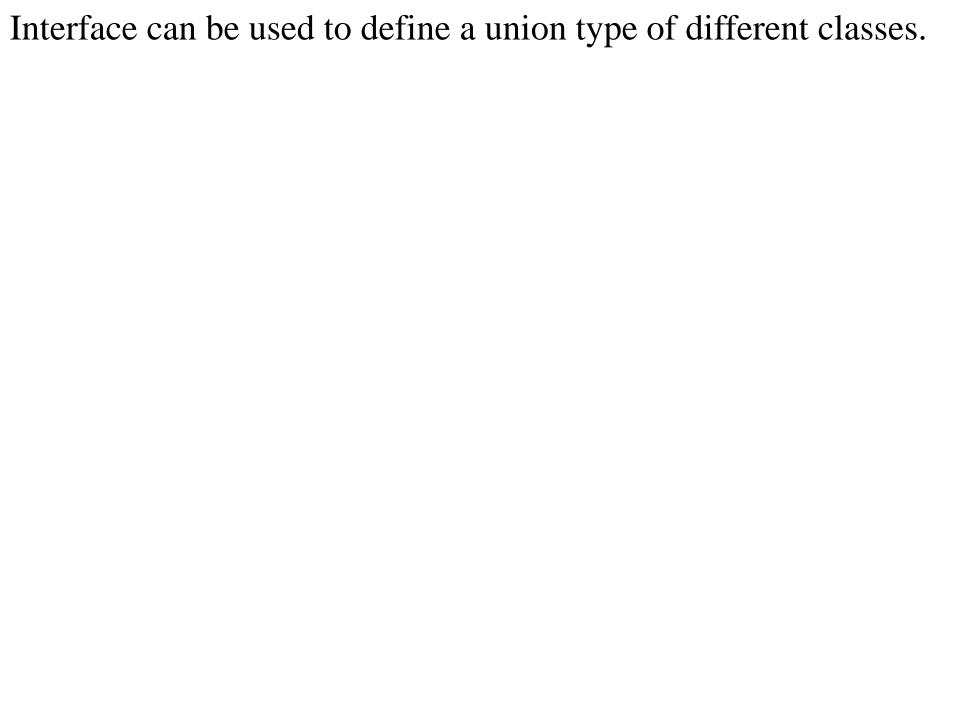
Interface: Abstraction/Polymorphism

Composition: Code Reuse

```
F(Rectangle r) {
...
}
```

F(new Square());

```
Class A {
 f(int a; int b);
 g
 h
Class AtoBAdapter {
 f(int a) { adaptee.f(a, 0); }
```



```
Class List<T> implements Iterable<Int>{
 Iterator<T> iterator() { ... new ListIterator(); ... }
Class ListIterator<T> implements Iterator<T> {
   hasNext() { ... }
   next() { ... }
   remove() { ... }
Class List<T> implements Iterable<T>, Iterator<T> {
  Iterator<T> iterator() { return this; }
  hasNext() { ... }
  next() { ... }
  remove() { ... }
```

```
A uses B
B uses C
Class A {
  Bb;
Class B {
  Cc;
Class A {
  IB b;
Interface IB { f(); g(); }
Class B {
  IC c;
```

What is computer science?

CS is "abstraction of mechanization"