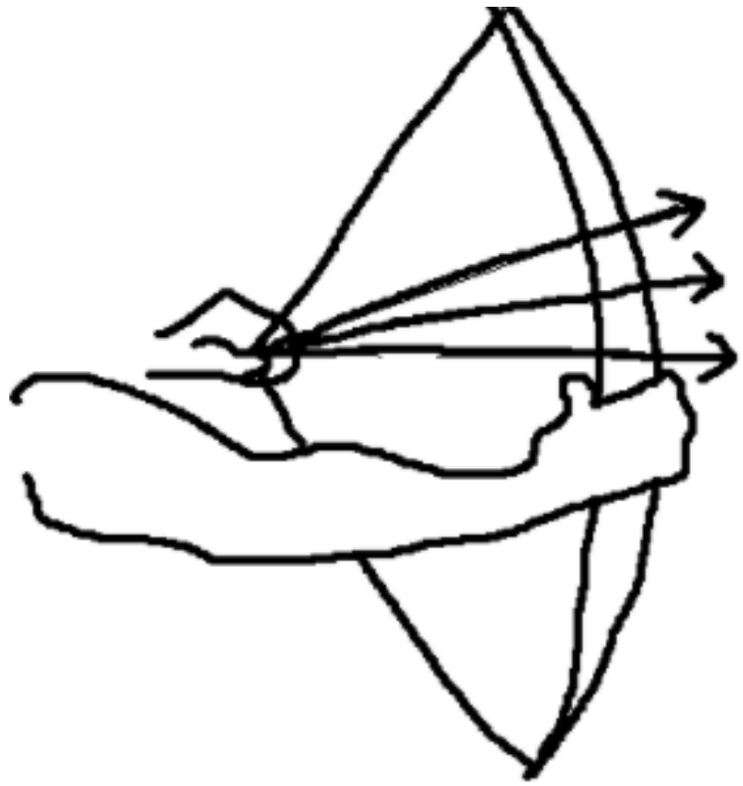
Overload

CSE2013-17F

slides are from CSE2013-16S at SKKU

Function Overloading

1) shares the same function name
2) different parameter lists




```
fun (int a)
fun (float a)
fun (int a, float b)
```

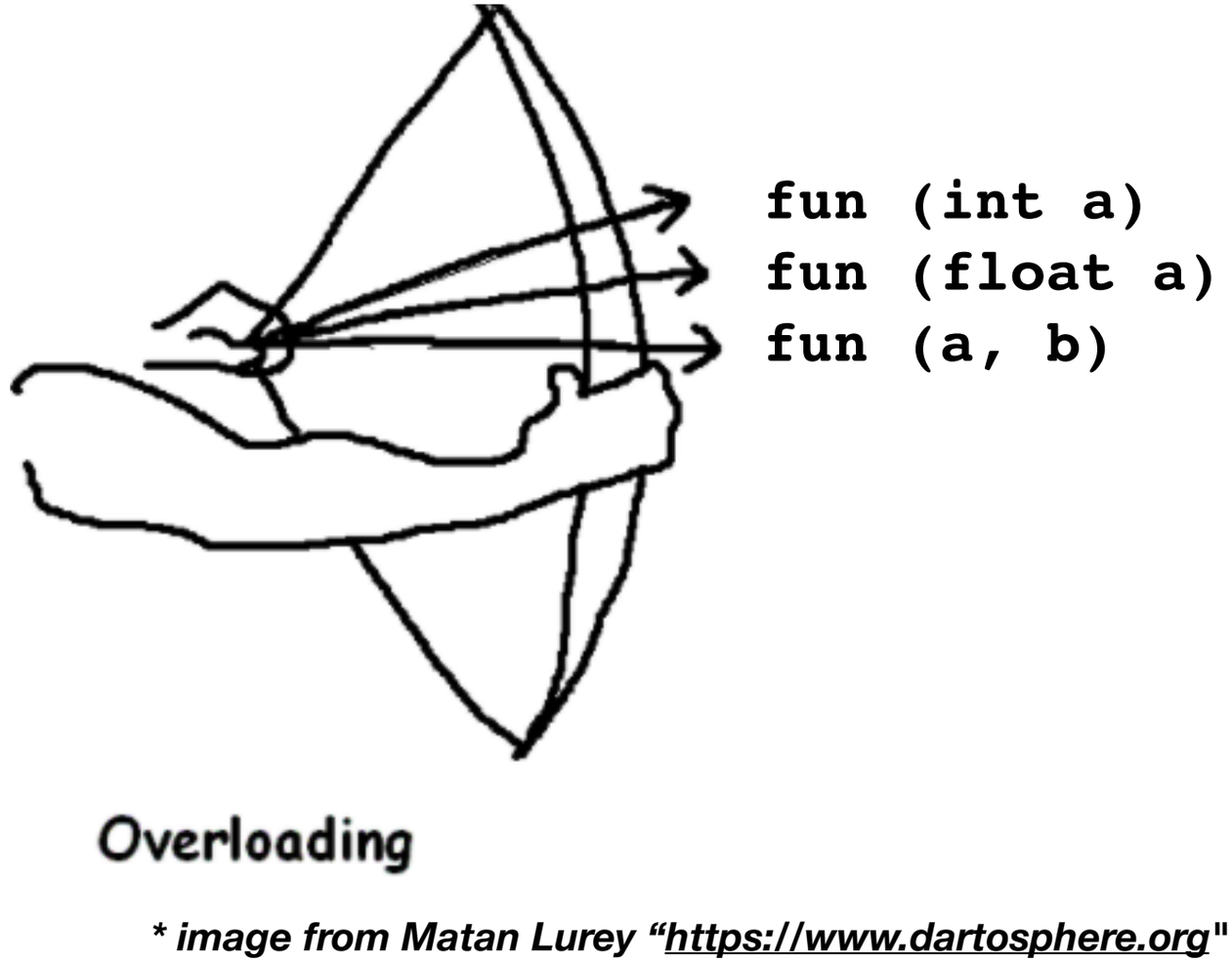
** image from Matan Lurey "https://www.dartosphere.org"*

* Compiler renames each function because machine code does not support function overloading

Function Overloading

```
void fun (int a);     // *1
void fun (float a);   // *2
void fun (int a, float b); // *3

int main( ) {
     int x = 123;
     float y = 12;

     fun (x); // *1
     fun (y); // *2

     fun (x, y);  // *3

     return 0;
}
```

1) shares the same function name
2) different parameter lists



** image from Matan Lurey "https://www.dartosphere.org"*

* Compiler renames each function because machine code does not support function overloading

Function Overloading

- Function overloading uses the same function name but different data types of parameter lists

- int Foo (int a), Foo (float b), Foo (int a, float b)

- ~~float Foo (int a), Boo (int a)~~

Operator Overloading

- Function overloading uses the same function name but different data types of parameter lists

- **Operator overloading - e.g. '*'**
  - *addr => returns value in the address *addr*
  - 2 * 5 => returns 2 x 5 = 10

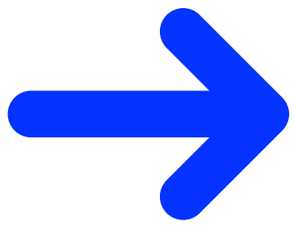
Operator Overloading

operator@(argument-list)

- **operator is a function**
- **@ is one of C++ operator symbols (+, -, =, *, /, ++, —, etc…)**

Operator Overloading

```
for (int i=0; i<100; i++)
     out[i] = objA.a[i] + objB.a[i];
```



```
out = objA + objB
```

```
class EX {
    int a[100];
public:
    int* operator+(const EX &Obj) {
        for (int i=0; i<100; i++) {
            out[i] = a[i] + Obj.a[i];
        }
        return out;
    }
};

int main( ) {
    EX A, B;
    int* tmp;
    …
    tmp = A.operator+(B);
    tmp = A + B;
    …
}
```

tmp = A + B + C + D + … is possible
A.operator+(B.operator+(C…))