Function Pointer

Stanley Yao
Computer Science Department
University of Arizona

---

Function

- A group of statements
- Starts from a certain address
- Function name itself is a pointer pointing to the start address of the function
- A function name contains many information
  - Point to the starting address of then function
  - Function pointer base type
  - Parameter type list
  - Return value type

---

Function, Variable and Pointer

```c
int (*p2)(void);
int *p1;
    ...
    i (of type int)
    ...
int foo(void) {
    <statements>
}
```

---

Function Pointer

```c
int (*func)(int foo);
```

```c
func: a function pointer
“func” points to a function that takes an int as parameter and returns int
The base type of “func” is a function that takes an int and returns an int
```

```c
Compare: int *func(int foo);
```

---

Using Function Pointers

```c
void foo (void) {
    printf("hello\n");
}
void main (void) {
    void (*ptr)(void);
    ptr = foo;
    (*ptr)(); // foo();
}
```

```c
ptr: a function pointer
*p2: the function to which “ptr” points to
(*p2)(): a call to the function
```

---

qsort

```c
void qsort(void *base, size_t nel, size_t width,
    int (*compar) (void *, void *));
```

```c
qsort() routine in the C library
The data items to be sorted can be of any type and size. How does qsort compare them during the sorting?
base: address of the array of elements to be sorted
nel: number of elements in the array
width: size (in bytes) of each element
compar: pointer to the compare function
```
Object

- Wrap data and functions in a structure
- Need to initialize the function pointers when initializing the object
  - struct __db dbp;
  - dbp.get = my_get_func;

```
struct __db {
  char *filename;
  int type;
  int (*get) (struct __db *dbp, void *key, void *data);
  int (*put) (struct __db *dbp, void *key, void *data);
};
```

Acknowledgement