

# COMP10002 Foundations of Algorithms

## Workshop Week8

Wenbin Cao

September 19, 2019

GitHub Repo: <https://github.com/AlanChaw/COMP10002-FoA>

# Outline

- C typedef
- struct

# typedef

- Give a type a new name
- Example

```
typedef int INTEGER;
```

```
INTEGER a, b;
```

```
a = 1;
```

```
b = 2;
```

```
#define LENGTH 100
```

```
typedef char word_t[LENGTH];
```

```
word_t word1 = "myword1";
```

```
word_t word2 = "myword2";
```

# struct

- Structure Define and Variable Declaration
- Structure Operations
  - copy
  - compare
  - input & output
- Structure with Functions
- Structure with Pointers
- Structure with Arrays

# struct

- Managing data via a single variable
- Define a struct, and declare variables

```
#define LENGTH 100
typedef struct {
    char name[LENGTH];
    char orbits[LENGTH];
    double distance;
    double mass;
    double radius;
} planet_t;

int main(){
    planet_t earth = {"Earth", "Sun", 149.6, 5.9736e+24, 6738.1}
    planet_t moon = {"Moon", "Earth", 0.3844, 7.349e+22, 1738.1}

    return 0;
}
```

## struct operations

- copy

```
planet_t another_planet = earth;
```

- compare not allowed

# struct operations

## - input

```
planet_t new_planet;  
  
scanf("%s %s %lf %lf %lf",  
      new_planet.name,  
      new_planet.orbits,  
      &new_planet.distance,  
      &new_planet.mass,  
      &new_planet.radius  
);
```

## - output

```
printf("name: %s\n", new_planet.name);  
printf("orbits: %s\n", new_planet.orbits);  
printf("distance: %lf\n", new_planet.distance);  
printf("mass: %lf\n", new_planet.mass);  
printf("radius: %lf\n", new_planet.radius);
```

## **struct, function, pointer**

- struct can be pass to a function, it is treated as a scalar variable
- the changes of struct in a function only changes the local variable
- to modify a struct in a function, pass its pointer instead
- functions can return structure variables



# structs and arrays

```
typedef struct {  
    char name[LENGTH];  
    int age;  
    int math_grade;  
    int english_grade;  
} student_t;  
  
typedef struct {  
    int class_num;  
    student_t all_students[MAX_NUM];  
} class_t;  
  
typedef struct {  
    char school_name[LENGTH];  
    class_t all_classes[MAX_NUM];  
} school_t;
```

# struct

- Structure Define and Variable Declaration
- Structure Operations
  - copy
  - compare
  - input & output
- Structure with Functions
- Structure with Pointers
- Structure with Arrays

# Exercise

8.2 Define a structure `vector_t` that could be used to store points in two dimensions  $x$  and  $y$  (such as on a map).

Then write a function `double distance(vector_t p1, vector_t p2)` that returns the Euclidean distance between `p1` and `p2`. If  $p1 = (x_1, y_1)$  and  $p2 = (x_2, y_2)$ , then the Euclidean distance between them is given by

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}.$$

Hint: You may use the functions in `<math.h>`

```
/* Returns x raised to the power of y. */  
double pow(double x, double y);  
  
/* Returns the square root of x. */  
double sqrt(double x);
```

# Assignment 1

- Submission
- General Questions