

C programming part 5

Struct revisited

defines a new type, CAL-DATE

```
struct CAL-DATE {  
    char month[11];  
    int day;  
    int year;  
} birthday, hireday;
```

declaration of structure

these are structures of type CAL-DATE

- Declare new variables of type CAL-DATE:

```
struct CAL-DATE labor-day;
```

```
struct CAL-DATE d[100]; // an array of elements of type  
                        // CAL-DATE
```

Example:

```
struct COMPLEX {  
    double real;  
    double imag;
```

```
};
```

```
struct COMPLEX a, b;
```

Access to the elements / members of a struct.

(2)

Example:

```
struct CAL-DATE cd, d[10];
```

```
cd.day = 24;
```

```
cd.year = 1991;
```

```
strcpy(cd.month, "September");
```

```
...
```

```
d[3].day = cd.day;
```

```
d[3].year = cd.year;
```

```
strcpy(d[3].month, cd.month);
```

- Operators `.` and `→` are used to refer to members of structures, unions, and classes.

Example:

```
int main ( ) {
```

```
    CAL-DATE d1, d2;
```

```
    CAL-DATE *pd;    // pd is a pointer to CAL-DATE!
```

```
    d1.day = 24;
```

```
    d1.year = 1991;
```

```
    strcpy(d1.month, "September");
```

```
    printf("Year of d1 is %d", d1.year);
```

```
    pd = &d1;
```

```
    printf("Date d1 is %s %d %d", pd->month, pd->day, pd->year);
```

```
    printf("Date d1 is %s %d %d", d1.month, d1.day, d1.year);
```

```
    printf("Date d1 is %s %d %d", (*pd).month, (*pd).day, (*pd).year);
```

```
}
```

- Struct is transferred to functions via pointers (or reference). (3)
- Avoid passing structs by value! (it was not even allowed)

```
void my-print(CAL-DATE *d) {  
    printf("Date is %s %d %d", d->month, d->day, d->year);  
}  
  
int main() {  
    CAL-DATE a;  
    a.day = 24;  
    a.year = 1991;  
    strcpy(a.month, "Sept.");  
    my-print(&a);  
}
```

- **Note:** if you pass by value the bytes of the struct are copied as the function parameter. Any changes inside the function will affect the local copy only.

Struct inside struct

```
struct CAL-DATE {  
    char month[11];  
    int day, year;  
};  
  
struct EMPLOYEE {  
    char name[64];  
    int id;  
    struct CAL-DATE birthday, hireday;  
};
```

Example:

(4)

```
struct EMPLOYEE e1;
```

```
e1.id = 101;
```

```
strcpy (e1.name, "Cris");
```

```
e1.birthday.day = 1;
```

```
e1.birthday.year = 2011;
```

```
strcpy (e1.birthday.month, "April");
```

...

```
printf ("Cris was born in the year of %d", e1.birthday.year);
```

```
struct EMPLOYEE *p;
```

```
p = &e1;
```

```
printf ("Year is %d", p->birthday.year);
```

Union

Enum

Assignment: read about Unions and Enums in the materials
from reading pointers on course website!