中国科学技术大学
UNIVERSITY OF SCIENCE AND TECHNOLOGY OF CHINA
Hefei,Anhui.230026 The People's Republic of China

Topic : OOP Object Oriented Programming

1. 历史 { 命令式
函数式
面向对象

2. OOP:

复数 complex

```
struct complex_t{

};
complex_new(...);
complex_add(..);
.....
```

C_1, C_2 ☆

使用OOP重新实现 →

```
class Complex{
    double x;
    double y;
    Complex(..){...}。构造函数
    add(C2){...}
    print(){...}
    distance(){...}
}
```

☆

C语言特性:
数据表示透明

客户端代码:
```
C1 = new Complex(...).
C2 = ....
C1.add(C2)
C1.print()
C2.distance( ).
```

用C语言实现面向对象编程

complex.h

```
typedef struct complex_t *complex_t;
struct complex_t{
    double x;
    double y;
    complex_t (*add)(complex_t, complex_t); //函数指针，中级
    void (*print)(complex_t);
    double (*distance)(complex_t);
}
```

complex.c

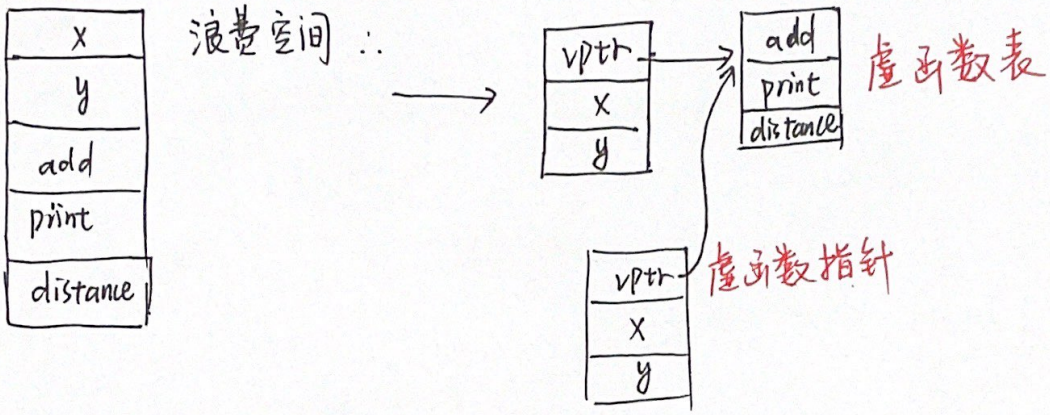
```
#include "complex.h"
complex_t new complex_new (double x, double y){
    complex_t tmp .....
```

客户端代码：

```
#define CALL(obj,f) obj->f(obj)
double d = CALL(c, distance);
```

元编程

作业：使用可变参数宏定义编写客户端代码.



↓ 如何实现?

complex.h

```
typedef struct complex_vtable *complex_vtable;

struct complex_vtable {
    complex_t (*add)(....
        ...
};

struct complex_t {
    complex_vtable vptr;
    double x;
    double y;
};
```

complex.c

```
struct complex_vtable vtable {
    .add = add;
    .print = ...
    .distance = distance
};
```

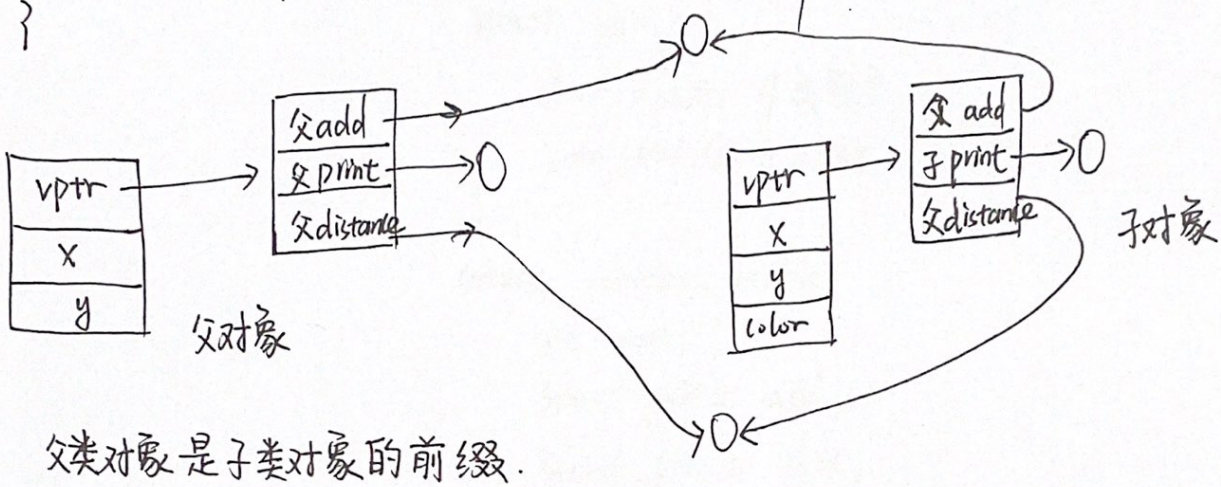
main.c

```
#define CALL(obj, f) obj->vptr->f(obj)
```

继承：

```
class ComplexColor extends Compelex{
    char *color;
    print(){...}  //重写print
}
```

子类.



父类对象是子类对象的前缀.

代码实现：

complex_color.h：

```
typedef  struct  complex_color_t *complex_color_t;
struct  complex_t{
    complex_vtable vptr;
    double x;
    double y;
    char *color;
}
.....
```

complex_color.c

反射和接口.

运行时获得代码的信息. eg: 虚函数表中函数的个数.
获取每个函数的名字.

```
struct pair_t{
    char *num; // 函数名
    void (*f)(); // 指针
}

struct complex_vtable{
    int num;
    struct pair_t add;
    struct pair_t print;
    ....
};

struct complex_vtable vtable={
    .num = 3,
    .add = {.name = "add", .f = add()}
    ......
};
```