以钟表为例。定义钟表数据和行为。

头文件 `clock.h`

```
#pragma once
class tagClock {
private:
    //数据
    int nHour;
    int nMinute;
    int nSecond;

public:
    //行为，函数声明
    int __stdcall SetTime(int nH, int nM, int nS);
};
```

实现头文件中定义的成员函数，源文件 `clock.cpp`

```
#include "stdafx.h"
#include "clock.h"

//函数定义，需要加上结构体名作用域
int tagClock::SetTime(int nH, int nM, int nS) {
    //this指针表示当前调用该函数的对象的地址
    this->nHour = nH;
    nMinute = nM;
    nSecond = nS;

    return 0;
}
```

实现所有函数后，主函数中调用 `main.cpp`

```
#include "stdafx.h"
```

```cpp
#include "clock.h"

//定义类中的函数指针
typedef int(__stdcall tagClock::*PFN_SetTimeClass)(int nH, int nM, int nS);
//参数传递：寄存器 栈
// thiscall --> 用于类的成员函数的调用约定， 编译器内部使用
// fastcall
// cdecl
// stdcall

int main()
{
    tagClock cl;

    //cl.nHour  = 1; //不可访问

    //第1种调用成员函数的形式
    cl.SetTime(1, 2, 3);

    //第2种调用成员函数的形式，通过函数指针
    PNF_SetTimeClass pfn_class;
    pfn_class = &tagClock::SetTime;
    (cl.*pfn_class)(11, 21, 31);

    //第3种调用成员函数的形式，通过对类对象的引用
    tagClock* pCL = &cl;
    //PNF_SetTimeClass pfn_class;
    //pfn_class = &tagClock::SetTime;
    (pCL->*pfn_class)(10, 20, 30);

    return 0;
}
```

- 第1种调用成员函数的形式

```
15
16      //第1种调用成员函数的形式
17      cl.SetTime(1, 2, 3);
18
19      //第2种调用成员函数的形式，通过函数指针
20      PFN_SetTimeClass pfn_class;
21      pfn_class = &tagClock::SetTime;  已用时间 <= 1ms
```

100 %

内存 1

地址: 0x006FF7E4

```
0x006FF7E4  01 00 00 00  ....
0x006FF7E8  02 00 00 00  ....
0x006FF7EC  03 00 00 00  ....
0x006FF7F0  cc cc cc cc  ????
```

监视 1

名称
cl
&cl

- 第2种调用成员函数的形式，通过函数指针

```
18
19      //第2种调用成员函数的形式，通过函数指针
20      PFN_SetTimeClass pfn_class;
21      pfn_class = &tagClock::SetTime;
22      (cl.*pfn_class)(11, 21, 31);
23
24      //第3种调用成员函数的形式，通过对函数对象的引用
25      tagClock* pCL = &cl;  已用时间<=1ms
```

100 %

内存 1

地址: 0x006FF7E4

```
0x006FF7E4  0b 00 00 00  ....
0x006FF7E8  15 00 00 00  ....
0x006FF7EC  1f 00 00 00  ....
0x006FF7F0  cc cc cc cc  ????
```

监视 1

名称	值
cl	{nHour=11 nMinute=21 nSecond=31 }
&cl	0x006ff7e4 {nHour=11 nMinute=21 nSecond=31 }

- 第3种调用成员函数的形式，通过对类对象的引用

```
24      //第3种调用成员函数的形式，通过对类对象的引用
25      tagClock* pCL = &cl;
26      //PNF_SetTimeClass pfn_class;
27      //pfn_class = &tagClock::SetTime;
28      (pCL->*pfn_class)(10, 20, 30);
29
30      return 0;  已用时间 <= 1ms
31  }
```

00 %

内存 1

地址: 0x006FF7E4

```
0x006FF7E4  0a 00 00 00  ....
0x006FF7E8  14 00 00 00  ....
0x006FF7EC  1e 00 00 00  ....
0x006FF7F0  cc cc cc cc  ????
0x006FF7F4  2a 11 84 ac  *.??
0x006FF7F8  0c f8 6f 00  .?o.
```

监视 1

名称	值
cl	{nHour=10 nMinute=20 nSecond=30 }
&cl	0x006ff7e4 {nHour=10 nMinute=20 nSecon