7_函数参数的扩展

函数默认参数值

> C++中可以在 `函数声明时为参数提供一个默认值`
> 当函数调用时 `没有提供参数的值`，则 `使用默认值`

1. 函数声明和函数定义**分开时**，参数的默认值必须在 `函数声明中指定`
2. 函数声明和函数定义**一体时**，且出现在 `main` 函数之前，
   此时**函数声明就是函数定义**，所以 **允许指定参数的默认值**

`lesson_7_1.cpp`

```
1  int mul(int x = 0);
2
3  int main(void)
4  {
5      int r;
6      r = mul();
7      printf("r = %d\n", r);
8
9      r = mul(2);
10     printf("r = %d\n", r);
11     return 0;
12 }
13
14 int mul(int x)
15 {
16     return x * x;
17 }
```

输出

```
1  r = 0
2  r = 4
```

问题

- A1：函数定义中是否可以出现参数的默认值?

  只允许以下两种情况指定参数的默认值，其它情况不允许：

  i. 函数声明和函数定义**分开时**，**只允许在函数声明中出现参数的默认值**

`lesson_7_a.cpp`

```c
#include <stdio.h>

int mul(int x=2);  // declare

int mul(int x)  // define
{
    return x * x;
}

int main(void)
{
    int r;
    r = mul();
    printf("r = %d\n", r);
}
```

ii. 函数声明和函数定义**一体时**，且出现在 `main` 函数之前，此时**函数声明就是函数定义**，所以**允许指定参数的默认值**

`lesson_7_b.cpp`

```c
#include <stdio.h>

int mul(int x=2)  // declare && define
{
    return x * x;
}

int main(void)
{
    int r;
    r = mul();
    printf("r = %d\n", r);
}
```

- A2：当函数声明和定义中的参数默认值不同时会发生什么?

> 看 A1

函数默认参数的规则

- 设计函数时，参数的默认值必须 `从右向左提供`，也就是需要使用默认值的参数都放在右边
- 函数调用时，**使用了默认值**，则后续的参数 `必须` 使用默认值

```c
int add(int x, int y = 1, int z = 2)
```

```
{
    return x + y + z;
}

add(0);  // x = 0, y = 1, z = 2，y使用了默认值，那么y后续的参数也使用了默认值
add(2, 3); // x = 2, y = 3, z = 2
add(2, 3, 1)  // x = 2, y = 3, z = 1
```

example lesson_7-2.cpp

输出

```
root@ubuntu:~/exp/DT_CPP/part01# ./a.out
1
3
6
```

example lesson_7-2-1.cpp

输出

```
root@ubuntu:~/exp/DT_CPP/part01# g++ lesson_7-2-1.cpp
lesson_7-2-1.cpp: In function ‘int main()’:
lesson_7-2-1.cpp:10:9: error: ‘z’ was not declared in this scope
   10 |     add(z=3);
      |         ^
```

函数占位参数与默认参数值

> 在C++中可以为函数提供 `占位参数`

- 占位参数 `只有参数类型声明`，而**没有参数名声明**
- 一般情况下，在函数体内部 `无法使用占位参数`

`lesson_7-3-1.cpp`

```
#include <stdio.h>

int func(int a, int);  // synax correct

int main(void)
{
    printf("%d\n", func(1, 2));
    return 0;
}

```

```
int func(int a, int)
{
    return a;
}
```

输出

```
1
```

意义

- 占位参数与默认参数结合起来使用
- `兼容C语言程序`中可能出现的不规范写法

`lesson_7-3-2.c`

```
#include <stdio.h>

int func();

int main(void)
{
    func();
    func(1, 2);

    return 0;
}

int func()
{
    return 1;
}
```

> 以上程序，在C编译器中编译是正常的，但在C++编译器编译是不通过的，且写法是不规范的。C++为了兼容，看 `lesson_7-3-2.cpp`

`lesson_7-3-2.cpp`

```
#include <stdio.h>

int func(int = 0, int = 0);  // match

int main(void)
{
    func();
    func(1, 2);

```

```
    return 0;
}

int func(int, int) // match
{
    return 1;
}
```

小结

- C++支持函数`参数的默认值`
- 如果函数`调用时没有提供参数值`，则使用默认值
- 参数的默认值必须`从右往左提供`
- 函数调用时`使用了默认值`，则后续参数必须使用默认值
- C++支持占位参数，`用于兼容C语言中的不规范写法`