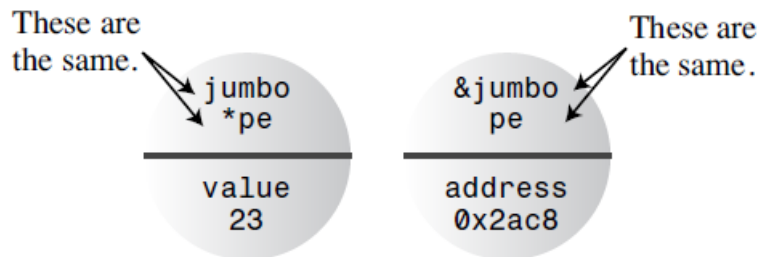


指针变量

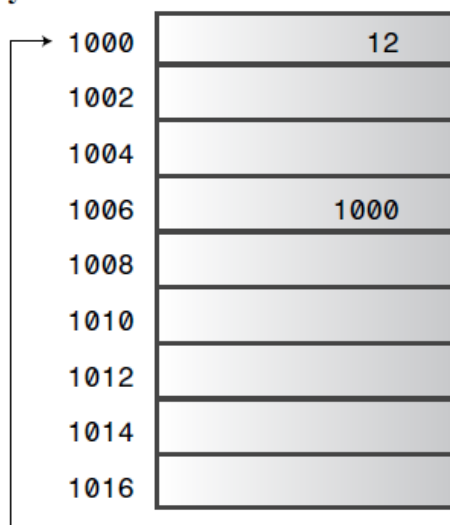
- 指针也是一种数据类型，指针变量也是一种变量
- 指针变量指向谁，就把谁的地址赋值给指针变量
- * 操作符操作的是指针变量指向的内存空间
- 使用 `sizeof()` 测量指针的大小，取决于操作系统

```
int jumbo = 23;  
int * pe = &jumbo;
```



```
1 char ch = 'A';  
2 int a = 1;  
3  
4 printf("%p %p\n", &ch, &a);
```

Memory address



Variable name

ducks
↑
birddog
points to
ducks
birddog

```
int ducks = 12;
```

creates ducks variable, stores
the value 12 in the variable

```
int *birddog = &ducks;
```

creates birddog variable, stores
the address of ducks in the variable

```
1 int a = 1;  
2 int* b = &a;  
3 *b = 2;  
4 cout << a << " " << *b << endl;
```

野指针：C++中创建指针时，计算机将分配用来存储地址的内存，但是不会分配用来存储指针所指向数据的内存。比如：p指针指向的空间是不确定的，通过间接操作修改了p指向空间的数据，很可能是操作系统中重要的数据，就发生不可预知的危险，因此，声明指针时一定要初始化。

```
1 int *p;
2 *p = 100;
3 cout << *p << endl;
```

空指针：野指针和有效指针变量保存的都是数值，为了标志此指针变量没有指向任何变量(空闲可用)，C语言中，可以把NULL赋值给此指针，这样就标志此指针为空指针，没有任何指针。

- `int *p = NULL`
- NULL是一个值为0的宏常量：`#define NULL((void *)0)`

万能指针 `void *`

有时候，一个指针根据不同的情况，指向的内容是不同类型的值，我们可以先不明确定义它的类型，只是定义一个无类型的指针，以后根据需要再用强制类型转换的方法明确它的类型。

```
1 #include <iostream>
2 using namespace std;
3
4 int a = 10;
5 double b = 3.5;
6 void* p;
7 int main(){
8     p = &a;
9     cout << *(int*)p << endl;
10    p = &b;
11    cout << *(double*)p << endl;
12    cout << *(long long*)p << endl;
13    return 0;
14 }
15
16 输出:
17 10
18 3.5
19 4615063718147915776
```

const修饰的指针变量

```
1 int a = 100, b = 200;
2
3 // 指向常量的指针
4 // 修饰*, 指针指向可以改变, 指针指向内存区域不能修改
5 const int * p1 = &a;
6 // *p1 = 111;
7 p1 = &b;
8 cout << *p1 << endl;
9
```

```

10 // 指针常量
11 // 修饰p2, 指针指向内存区域可以改变, 指针指向不可以改变
12 int * const p2 = &a;
13 // p2 = &b;
14 *p2 = 222;
15 cout << *p2 << endl;

```

指针和数组

```

1  #include <iostream>
2  #include <cstdio>
3  using namespace std;
4
5  int a[5];
6  int* pa = a;
7
8  int main() {
9
10     for(int i = 0; i < 5; i++)
11         scanf("%d", a + i);
12
13     for(int i = 0; i < 5; i++)
14         printf("a[%d]=%d\n", i, *(a + i));
15
16     return 0;
17 }

```

```

1  #include <iostream>
2  #include <cstdio>
3  using namespace std;
4
5  int a[5];
6  int* pa = a;
7
8  int main() {
9
10     for(int i = 0; i < 5; i++)
11         scanf("%d", a + i);
12
13     for(int i = 0; i < 5; i++){
14         printf("%d ", *pa);
15         pa++;
16     }
17
18     return 0;
19 }

```

指针数组

数组的每个元素都是指针类型。

公众号：黑猫编程

网址：<https://noi.hioier.co>

```

1  #include <iostream>
2  #include <cstdio>
3  using namespace std;
4
5  int main() {
6
7      int a = 1, b = 2, c = 3;
8      int* p[] = {&a, &b, &c};
9
10     for(int i = 0; i < sizeof(p) / sizeof(p[0]); i++)
11         cout << *p[i] << " ";
12
13     return 0;
14 }

```

多重指针

```

1  #include <cstdio>
2
3  int a = 10;
4  int* p;
5  int** pp; // 定义双重指针
6  int*** ppp; // 定义三重指针
7
8  int main() {
9      p = &a; // 将p指向a
10     pp = &p; // 将pp指向p
11     ppp = &pp; // 将ppp指向pp
12     printf("a=%d=%d=%d\n", *p, **pp, ***pp);
13     return 0;
14 }

```

引用

引用是给变量起别名，比指针更加简洁。

```

1  #include <iostream>
2  #include <cstdio>
3  using namespace std;
4
5  int main() {
6
7      // 引用的本质是常指针，因此必须初始化
8      int a = 10;
9      int& b = a; // int* const b = &a;
10
11     b = 20; // *b = 20;
12
13     cout << a << " " << b << endl;
14
15     return 0;

```

```
16 }
```

```
1  #include <iostream>
2  #include <cstdio>
3  using namespace std;
4
5  int f(int& x){
6      x *= 2;
7      return x * 3;
8  }
9
10 int main() {
11
12     int a = 10;
13     cout << f(a) << " " << a << endl;
14
15     return 0;
16 }
```

数组引用

```
1  int a[5] = { 1, 2, 3, 4, 5 };
2  int(&aref)[5] = a;
3
4  aref[0] = 6;
5
6  for (int i = 0; i < 5; i++) cout << a[i] << " "; // 6 2 3 4 5
```

函数参数实现变量交换

```
1  #include <iostream>
2  #include <cstdio>
3  using namespace std;
4
5  void swap1(int a, int b){
6      int t = a;
7      a = b;
8      b = t;
9      cout << "函数内部: " << a << " " << b << endl;
10 }
11
12 void swap2(int* a, int* b){
13     int t = *a;
14     *a = *b;
15     *b = t;
16 }
17
18 void swap3(int& a, int& b){
19     int t = a;
```

```

20     a = b;
21     b = t;
22 }
23
24 int main() {
25
26     int a = 1, b = 2;
27
28     // swap1(a, b);
29     // swap2(&a, &b);
30     swap3(a, b);
31
32     cout << "函数外部: " << a << " " << b << endl;
33
34     return 0;
35 }

```

动态开辟空间

```

1  #include <iostream>
2  #include <cstdio>
3  using namespace std;
4
5  /*
6  const int N = 1e8 + 10;
7  int a[N];
8  */
9
10 int main() {
11
12     int* p = new int(3);
13     cout << *p << endl;
14
15     delete p;    // 释放空间
16
17     cout << *p << endl;
18
19     int* p2 = new int;
20     cout << *p2 << endl;
21
22     return 0;
23 }

```

```

1  #include <iostream>
2  #include <cstdio>
3  using namespace std;
4
5  int main() {
6
7     int n;
8     cin >> n;
9     int* a = new int[n + 1];

```

```

10
11     for(int i = 1; i <= n; i++)
12         cin >> a[i];
13
14     for(int i = 1; i <= n; i++)
15         cout << a[i] << " ";
16
17     delete [] a;
18
19     return 0;
20 }

```

数组做函数参数

```

1  #include <iostream>
2  #include <cstdio>
3  using namespace std;
4
5  int n;
6  int a[110];
7
8  void f(const int a[]){
9      // a[1] *= 2;
10     for(int i = 1; i <= n; i++)
11         cout << a[i] << " ";
12 }
13
14 void f2(const int* a){
15     // a[1] *= 2;
16     for(int i = 1; i <= n; i++)
17         cout << a[i] << " ";
18 }
19
20 int main() {
21
22     cin >> n;
23
24     for(int i = 1; i <= n; i++)
25         cin >> a[i];
26
27     f(a);
28
29     cout << endl << "-----" << endl;
30
31     for(int i = 1; i <= n; i++)
32         cout << a[i] << " ";
33
34     return 0;
35 }

```

字符串指针

```
1  #include <iostream>
2  #include <cstdio>
3  #include <cstring>
4  using namespace std;
5
6  int main() {
7
8      // char str[] = "hello cat";
9      const char * str = "program";
10
11     printf("%s\n", str);
12     cout << str << endl;
13
14     str = "hello";
15
16     printf("%s\n", str);
17     cout << str << endl;
18
19     int len = strlen(str);
20     for(int i = 0; str[i]; i++)
21         cout << *(str + i);
22
23     // str[0] = 'z';
24
25     return 0;
26 }
```