The `const` Keyword with Pointers

**Pointer to Constant**

```c
1:  const int i1 = 100;
2:  const int i2 = 200;
3:  const int *pi = &i1;          /* pi is a "pointer to a constant." */
4:  i1 = 300;                     /* Error. Cannot change i1. */
5:  i2 = 400;                     /* Error. Cannot change i2. */
6:  pi = &i2;                     /* OK. */
7:  *pi = 500;                    /* Error. Cannot change *pi. */
```

**Constant Pointer**

```c
1:  int i1 = 100;
2:  int i2 = 200;
3:  int *const pi = &i1;          /* pi is a "constant pointer." */
4:  i1 = 300;                     /* OK. */
5:  i2 = 400;                     /* OK. */
6:  pi = &i2;                     /* Error. Cannot change pi. */
7:  *pi = 500;                    /* OK. */
```

**Constant Pointer to Constant**

```c
1:  const int i1 = 100;
2:  const int i2 = 200;
3:  const int *const pi = &i1;    /* pi is a "constant pointer to a constant." */
4:  i1 = 300;                     /* Error. Cannot change i1. */
5:  i2 = 400;                     /* Error. Cannot change i2. */
6:  pi = &i2;                     /* Error. Cannot change pi. */
7:  *pi = 500;                    /* Error. Cannot change *pi. */
```
Disallowed Mismatch

1:  const int i1 = 100;
2:  const int i2 = 200;
3:  int *pi = &i1;        /* Error. Subversive. 
   Subsequently changing *pi would change i1. */

Disallowed Mismatch in Function Calls

1:  void f(int *pi)
2:  {
3:     ...
4:  }

... 

5:  const int i1 = 5;
6:  const int *pi2 = &i1;
7:  f(pi2);               /* Error. Subversive. 
   If f() changes *pi, then *pi2 also would change. */

Allowed Mismatch

1:  int i1 = 100;
2:  int i2 = 200;
3:  const int *pi = &i1;  /* OK, even though subsequently changing i1 would change *pi. */
4:  i1 = 300;             /* OK. Also changes *pi. */
5:  i2 = 400;             /* OK. */
6:  pi = &i2;             /* OK, even though subsequently changing i2 would change *pi. */
7:  *pi = 500;            /* Error. Cannot change *pi. */

Allowed Mismatch in Function Calls

1:  void f(const int *pi)
2:  {
3:     ...
4:  }

... 

5:  int i1 = 5;
6:  int *pi2 = &i1;
7:  f(pi2);               /* OK. *pi2 is protected against accidental change by f(). */