Princeton University
COS 217: Introduction to Programming Systems
The "const" Keyword with Pointers

Pointer to Constant

```c
const int iFirst = 100;
const int iSecond = 200;
const int *piThird = &iFirst;       /* piThird is a "pointer to a constant." */
iFirst = 300;                       /* Error. Cannot change iFirst. */
iSecond = 400;                      /* Error. Cannot change iSecond. */
piThird = &iSecond;                 /* OK. */
*piThird = 500;                     /* Error. Cannot change *piThird. */
```

Constant Pointer

```c
int iFirst = 100;
int iSecond = 200;
int *const piThird = &iFirst;       /* piThird is a "constant pointer." */
iFirst = 300;                       /* OK. */
iSecond = 400;                      /* OK. */
piThird = &iSecond;                 /* Error. Cannot change piThird. */
*piThird = 500;                     /* OK. */
```

Constant Pointer to Constant

```c
const int iFirst = 100;
const int iSecond = 200;
const int *const piThird = &iFirst; /* piThird is a "constant pointer to a constant." */
iFirst = 300;                       /* Error. Cannot change iFirst. */
iSecond = 400;                      /* Error. Cannot change iSecond. */
piThird = &iSecond;                 /* Error. Cannot change piThird. */
*piThird = 500;                     /* Error. Cannot change *piThird. */
```
Disallowed Mismatch

```c
const int iFirst = 100;
const int iSecond = 200;
int *piThird = &iFirst;  /* Error. Subversive. Subsequently changing *piThird */
                        /* would change iFirst. */
```

Disallowed Mismatch in Function Calls

```c
void f(int *piThird)
{
    ...
}
...
const int iFirst = 5;
const int *piSecond = &iFirst;
f(piSecond);  /* Error. Subversive. If f changes *piThird, then
               *piSecond also would change. */
```

Allowed Mismatch

```c
int iFirst = 100;
int iSecond = 200;
const int *piThird = &iFirst; /* OK, even though subsequently changing iFirst would */
                              /* change *piThird. */
iFirst = 300;  /* OK. Also changes *piThird. */
iSecond = 400;  /* OK. */
piThird = &iSecond; /* OK, even though subsequently changing iSecond would */
                   /* change *piThird. */
*piThird = 500;  /* Error. Cannot change *piThird. */
```

Allowed Mismatch in Function Calls

```c
void f(const int *piThird)
{
    ...
}
...
int iFirst = 5;
int *piSecond = &iFirst;
f(piSecond);  /* OK. *piSecond is protected against accidental change by f. */
```