Pointers – Array dereferencing, type independence

- Array subscript operator ([]) is a form of indexed dereferencing.
- C++ pointers are strictly typed; can't have an **int** pointer point to a **char** pointer, for example.

```
int myInt = 42;
int* pMyInt = &myInt;

char c = 'c'
char* pMyChar = &c;

pMyInt = pMyChar;
```

Pointers – Typecasting

- We *can, however, cast one data type to another:*

```
int myInt = 42;
int* pMyInt = &myInt;

char c = 'c'
char* pMyChar = &c;

// pMyInt = pMyChar;
pMyInt = reinterpret_cast< int* >( pMyChar );
```

- There are several typecasts you can do. *reinterpret_cast* could be interpreted as the *most dangerous*. *static_cast* is safer, but there are situations in which it won't do.
- Old C-style casts like *pMyInt = (int*)pMyChar* will still work, though. These are like reinterpret_cast<>.

Pointers and new

- Of course, pointers start to become really useful with *new, when we allocate memory on the heap:*

```
int* pMyIntAry = new int[1000 ];

pMyIntAry[ 25 ] = 1001;
// ... do some interesting stuff here

delete[] pMyIntAry;
```

Memory layout: