Pointers

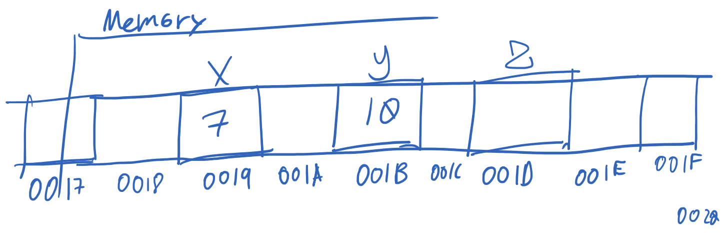
Wednesday, August 21, 2019 4:58 PM

C++ Pointers.

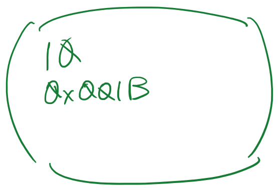
A pointer is a data type that stores memory addresses.

Example:

```
int x, y;
float z;
x = 7;
y = x + 3;
cout << y;
cout << &y;
```

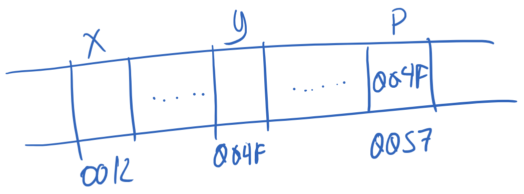


the & address-of operator.



Example:

```
int x
int y;
x = &y; ⊗
int *p;
p = &y; ✓
```



```
float *p;
string *q;
char *r;
```

Pointers carry type.

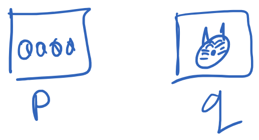
```
int x;
char *p;
p = &x; ⊗
float *q
q = &x; ⊗
```

Example:

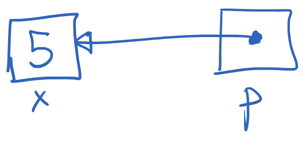
```
int *p1;
string *s;
float *p;
p = NULL;
```

The special Value NULL

```
char *p = NULL;
char *q;
```



```
int x = 5;
int *p = NULL;
```



Example:

```
int x;
const int y = 3;
int *p;
p = &y;
```

int *p = NULL;
p = &x;
*p;

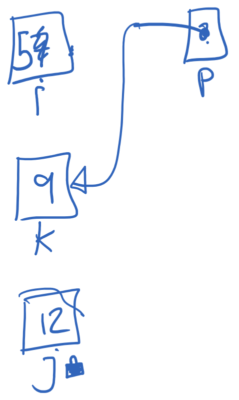
The dereference operator: *
Evaluates the the value of the variable pointed by a pointer:

*p;

Example:

```
int i = 7;
int *p;
p = &i;
cout << *p;
*p = 5;
cout << i;
```

Int k = 9;
p = &k;
const int j = 12;
p = &j;



Example:

```
int x;
int *p;
const int *q;
int const *r = &x;
const int const *s = &x;
```

int *p; — pointer to integer
const int *q; — Pointer to integer constant: q=&j
int const *r = &x; — constant pointer to integer.
const int const *s = &x; — constant pointer to integer constant

pointes as parameters;

int foo (int* p);

pointers as return values;

char* foo (int x);

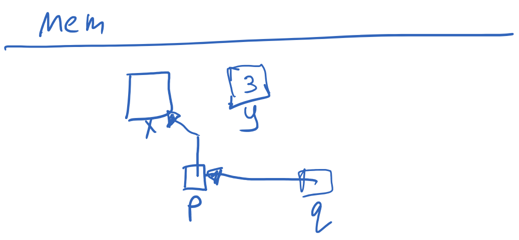
arrays of pointers:

int * z[10];

pointers in classes

class Chicken
{
int age:
int * whatever:
}

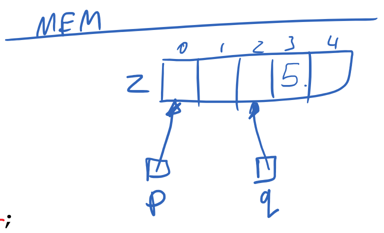
Chicken* p:



Exercise:
```
int x, y;
int *p;
p = &y;
*p = 7;
y = 5 * *p;
int **q;
q = &p;
**q = 3;
*q = &x;
```

• Pointers to Arrays:

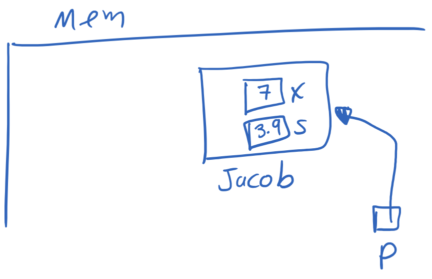
Example:
```
int z[5];
int *p;
p = &z[0];⇐
p = z;
p[3] = 42;
int *q;
q = &z[2];  ~~q = z[2];~~
q[1] = 5;
```



Example:
;

• Pointers to Classes / Structs

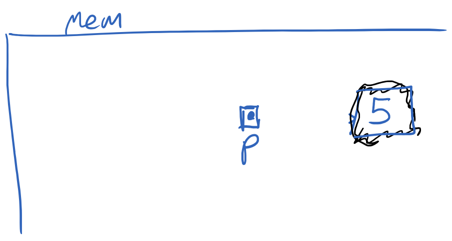
Example:
```
class Pineapple
{
public:
    int x;
    float s;
};

Pineapple Jacob;
Pineapple *p = &Jacob;
*p.x = 7;
(*p).x = 7;
p->x = 7;
p->s = 3.9;
```



← The -> operator

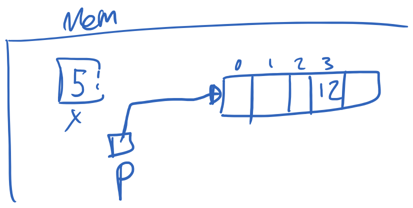
• The new and delete operators

Example:
```
int *p;
p = new int;
*p = 5;
…
…
delete p;
p = NULL;
```



= on Arrays

Example:
```
int *p;
int x;
cin >> x;
p = new int[x];
p[3] = 12;
x = 678;
……
delete [] p;
p = NULL;
```



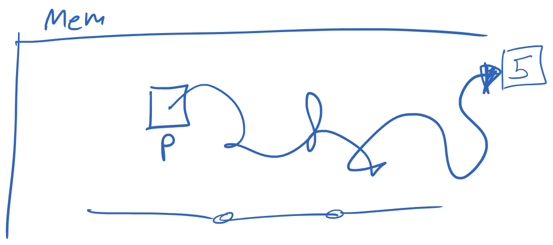
• Problems with Pointers

Dangling Pointer :- using a pointer with an invalid address / value

Memory Leak :- when dynamic variables become unreachable

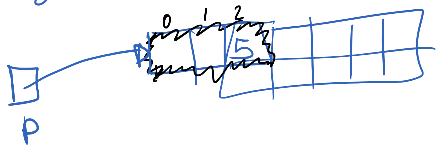
Dangling Pointers

Example:
```
int *p;
*p = 5;
```
Segmentation fault



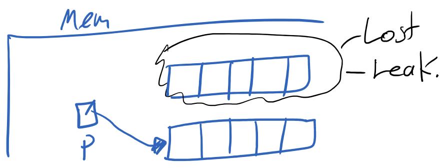
Example:
```
int *p = NULL;
*p = 5;
```

Example:
```
int *p = new int[3];
delete [] p;
p[2] = 5;
```

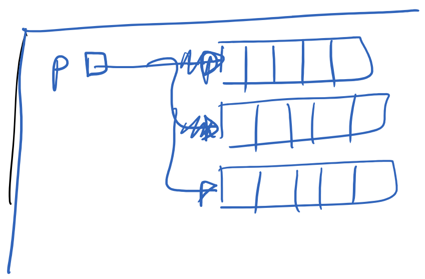


Memory Leak:

Example:
```
int *p = new int[5];
p = new int[5];
```

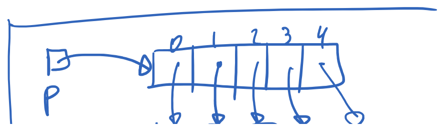


Example:
```
int *p;
while ( !done ) {
    p = new int[5];
}
```



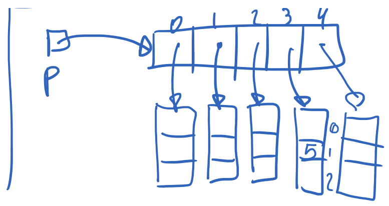
• pointers and 2D Arrays:

Example:
```
int **p;
p = new int*[5];
```



Example:

```
int **p;
p = new int*[5];
for( int i=0; i<5; i++ )
    p[i] = new int[3];

p[3][1] = 5;
```



```
for( int i=0; i<5; i++ )
    delete [] p[i];
delete [] p;
p = NULL;
```