Object Oriented Programming

Lecture

Array of Objects

Array of Objects

- Array of objects can only be created if an object can be created without supplying an explicit initializer
- There must always be a default constructor if we want to create array of objects

Example

```
class Test
{
   int i;
public:
};

int main(){
 Test array[2];  // OK
}
```

Example

```
class Test
{
   int i;
public:
 Test();
};

int main(){
 Test array[2]; // OK
}
```

Example

```
class Test
{
   int i;
public:
 Test(int x) {i=x;}
};

int main(){
 Test array[2]; // Error
}
```

Example

```
class Test
{
   int i;
public:
 Test(int x)  {i=x};
};

int main(){
 Test array[2]= {1,2};
}
```

Array[0].i = 1
Array[1].i = 2

Explicit initializer

Example

```
class Test
{
   int i,j;
public:
 Test(int x, int y) {i=x; j=y;};
};

int main(){
 Test array[2]= {{1,1},{2,2}};
}
```

Array[0].i = 1
Array[1].i = 2

Array[0].j = 1
Array[1].j = 2

Example

```
class Test
{
   int i,j;
public:
 Test(int x, int y) {i=x; j=y;};
};

int main(){
 Test a(1,1), b(2,2);
 Test array[2]= {a,b};
}
```

Array[0].i = 1
Array[1].i = 2

Array[0].j = 1
Array[1].j = 2

Pointer to Objects

Pointer to Objects

- Pointer to objects are similar as pointer to built-in types
- They can also be used to dynamically allocate objects

Example

```
class Rectangle
{
    int width, height;
public:
    Rectangle(int x=0, int y=0);
    int get_width();
    int get_height();
};

Rectangle::Rectangle(int x = 0, int y = 0)
{
    width = x;
    height = y;
}

int Rectangle::get_width()
{
    return width;
}

int Rectangle::get_height()
{
    return height;
}
```

```
int main()
{

    Rectangle obj;
    Rectangle* ptr;
    ptr = &obj;
    ptr->get_width();
    return 0;

}
```

```
Rectangle* ptr=obj;
ptr->getwidth();
(*ptr).getwidth;
obj.getwidth;
```

Thanks a lot