Create an Copied Object

- The **copy constructor** is a constructor which creates an object by initializing it with an object of the same class, which has been created previously.
- The copy constructor is used to:
  - Initialize one object from another of the same type.
  - Copy an object to pass it as an argument to a function.
  - Copy an object to return it from a function.

Create an Clone Object

- If a copy constructor is not defined in a class, the compiler itself defines one.
- If the class has pointer variables which is used for dynamic memory allocations, copy constructor must to be defined by a programmer.
- The most common form of copy constructor is shown here:

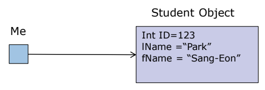
```
classname (const classname &obj)
{
   // body of constructor
}
```

Create an Clone Object

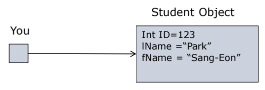
```
// example1.cpp: default constructor will be used

#include <iostream>
using namespace std;

struct Student{
   int iD;
   char fName[20];
   char lName[20];
};

int main()
{
   Student Me ={123, "Sang-Eon", "Park"};
   Student You = Me;
   Me.iD = 234;
   cout <<Me.fName <<',' << Me.lName << Me.iD <<endl;
   cout <<You.fName<<','<< You.lName << You.iD <<endl;
   return 0;
}
```

Create an Clone Object

```
Student Me ={123, "Sang-Eon", "Park"};
```

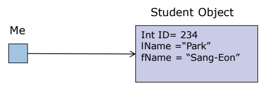


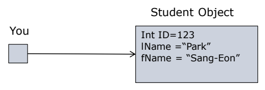
```
Student You = Me;
```



Create an Clone Object

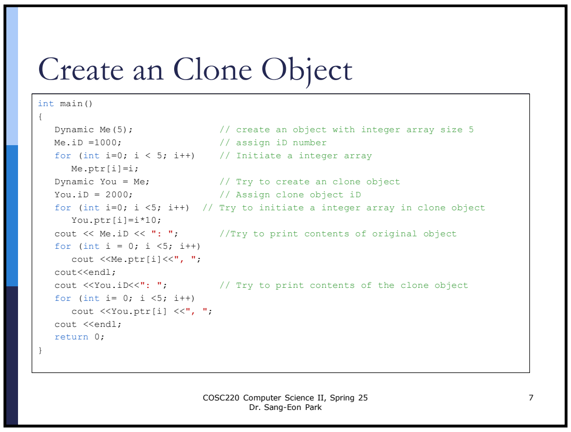
```
Me.iD = 234;
```





Create an Clone Object

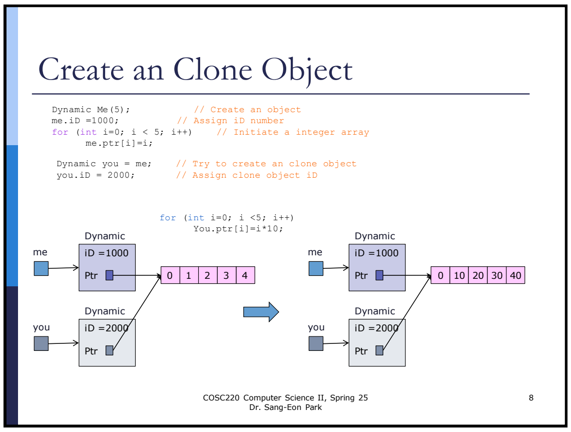
```
// example2.cpp without defining copy constructor

#include<iostream>
using namespace std;

class Dynamic{
public:
   int iD;
   int *ptr;            //for dynamic memory allocation
   Dynamic (int l);     // constructor
};

Dynamic::Dynamic(int l) // constructor
{
   ptr = new int(l);    //create a integer array size l
}
```



Create an Clone Object

```
int main()
{
   Dynamic Me(5);                // create an object with integer array size 5
   Me.iD =1000;                  // assign iD number
   for (int i=0; i < 5; i++)     // Initiate a integer array
      Me.ptr[i]=i;
   Dynamic You = Me;             // Try to create an clone object
   You.iD = 2000;                // Assign clone object iD
   for (int i=0; i <5; i++)  // Try to initiate a integer array in clone object
      You.ptr[i]=i*10;
   cout << Me.iD << ": ";        //Try to print contents of original object
   for (int i = 0; i <5; i++)
      cout <<Me.ptr[i]<<", ";
   cout<<endl;
   cout <<You.iD<<": ";          // Try to print contents of the clone object
   for (int i= 0; i <5; i++)
      cout <<You.ptr[i] <<", ";
   cout <<endl;
   return 0;
}
```

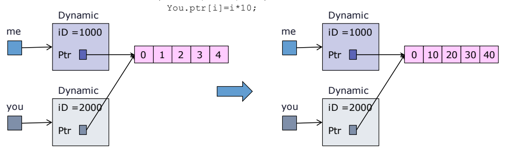
COSC220 Computer Science II, Spring 25
Dr. Sang-Eon Park 7



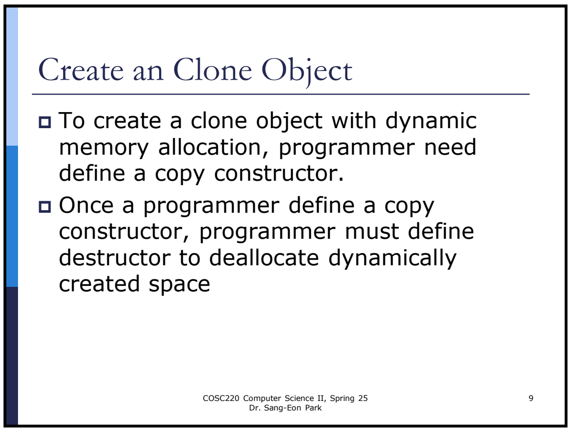
Create an Clone Object

```
Dynamic Me(5);              // Create an object
me.iD =1000;            // Assign iD number
for (int i=0; i < 5; i++)     // Initiate a integer array
      me.ptr[i]=i;

 Dynamic you = me;      // Try to create an clone object
 you.iD = 2000;         // Assign clone object iD
```

```
for (int i=0; i <5; i++)
      You.ptr[i]=i*10;
```



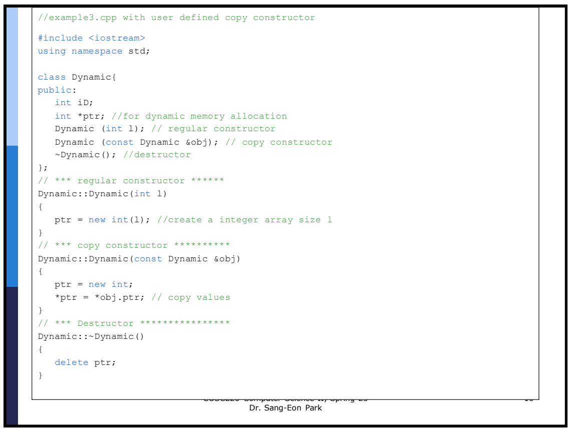
COSC220 Computer Science II, Spring 25
Dr. Sang-Eon Park 8



Create an Clone Object

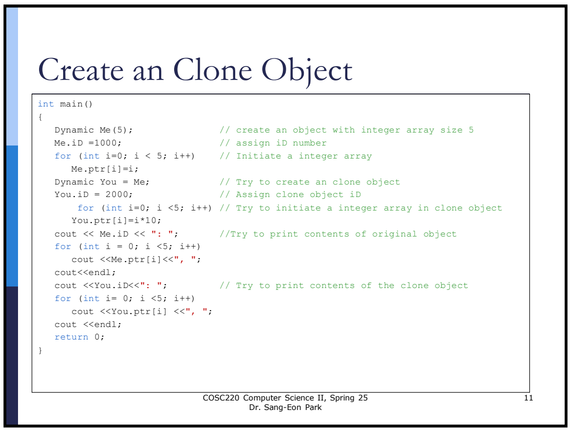
- To create a clone object with dynamic memory allocation, programmer need define a copy constructor.
- Once a programmer define a copy constructor, programmer must define destructor to deallocate dynamically created space

COSC220 Computer Science II, Spring 25
Dr. Sang-Eon Park 9



```
//example3.cpp with user defined copy constructor

#include <iostream>
using namespace std;

class Dynamic{
public:
   int iD;
   int *ptr; //for dynamic memory allocation
   Dynamic (int l); // regular constructor
   Dynamic (const Dynamic &obj); // copy constructor
   ~Dynamic(); //destructor
};
// *** regular constructor ******
Dynamic::Dynamic(int l)
{
   ptr = new int(l); //create a integer array size l
}
// *** copy constructor **********
Dynamic::Dynamic(const Dynamic &obj)
{
   ptr = new int;
   *ptr = *obj.ptr; // copy values
}
// *** Destructor ****************
Dynamic::~Dynamic()
{
   delete ptr;
}
```

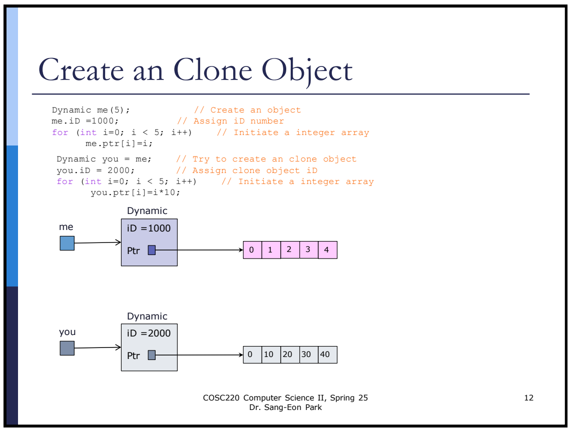
Dr. Sang-Eon Park



Create an Clone Object

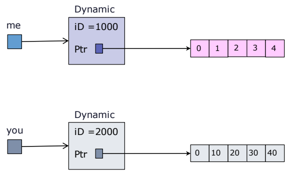
```
int main()
{
   Dynamic Me(5);                // create an object with integer array size 5
   Me.iD =1000;                  // assign iD number
   for (int i=0; i < 5; i++)     // Initiate a integer array
      Me.ptr[i]=i;
   Dynamic You = Me;             // Try to create an clone object
   You.iD = 2000;                // Assign clone object iD
       for (int i=0; i <5; i++) // Try to initiate a integer array in clone object
      You.ptr[i]=i*10;
   cout << Me.iD << ": ";        //Try to print contents of original object
   for (int i = 0; i <5; i++)
      cout <<Me.ptr[i]<<", ";
   cout<<endl;
   cout <<You.iD<<": ";          // Try to print contents of the clone object
   for (int i= 0; i <5; i++)
      cout <<You.ptr[i] <<", ";
   cout <<endl;
   return 0;
}
```

COSC220 Computer Science II, Spring 25
Dr. Sang-Eon Park 11

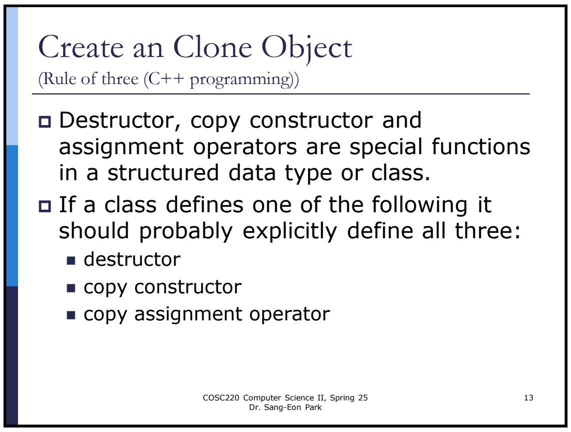


Create an Clone Object

```
Dynamic me(5);              // Create an object
me.iD =1000;            // Assign iD number
for (int i=0; i < 5; i++)     // Initiate a integer array
      me.ptr[i]=i;
 Dynamic you = me;      // Try to create an clone object
 you.iD = 2000;         // Assign clone object iD
 for (int i=0; i < 5; i++)     // Initiate a integer array
      you.ptr[i]=i*10;
```



COSC220 Computer Science II, Spring 25
Dr. Sang-Eon Park 12



Create an Clone Object
(Rule of three (C++ programming))

- Destructor, copy constructor and assignment operators are special functions in a structured data type or class.
- If a class defines one of the following it should probably explicitly define all three:
  - destructor
  - copy constructor
  - copy assignment operator

COSC220 Computer Science II, Spring 25
Dr. Sang-Eon Park

13