Given class Array as below that describes an array allocated in Heap memory.

class Array {

public:

Array(int size, int initValue);

~Array();

void setAt(int idx, int value);

int getAt(int idx);

int & operator[] (int idx);

Array(const Array & other); // Copy constructor

Array & operator= (const Array & other); // Copy assignment operator

private:

int size;

int \* p;

};

There are 2 attributes in class Array:

* p: a pointer contains the address of first element of allocated memory.
* size: number of elements of array.

Requirement: Implement following methods:

* Method Array(int size, int initValue): constructor, assigns size to the number of elements of the array; initializes a 1-dimensional array in the heap and stores the address of the first element of the array in the variable p.
* Method ~Array(): destructor, recovers the allocated Heap memory.
* Method print(): prints each element of array separated by 1 space, then print 1 new line character after last element.
* setAt(int idx, int value): set value to element at position ***idx***. If idx has an invalid value (idx < 0 or idx >= size), **throw -1**.
* getAt(int idx): returns value at position idx. If idx has an invalid value (idx < 0 or idx >= size), **throw -1**.
* operator [] (int idx): overload operator [] to do: read (get) the value of element at position idx, and write (set) a new value to the element at position idx. For instance: read: x = a[2], write: a[2] = 5, for a is an object of class Array. if idx has an invalid value (idx < 0 or idx >= size), then throw -1;
* Copy Constructor: initialize new object based on data of provided object.
* Copy Assignment operator: assigns the value of the current object's attributes to the value of the provided object.

**For example:**

| **Test** | **Result** |
| --- | --- |
| Array a1(5, 0);  a1.print(); | 0 0 0 0 0 |

| **Test** | **Result** |
| --- | --- |
| Array \* a7 = new Array(10, 2);  a7->print();  try {  a7->setAt(-1, 99);  a7->print();  }  catch (int exp) {  cout << "Exception: " << exp << endl;  }  try {  a7->setAt(10, 99);  a7->print();  }  catch (int exp) {  cout << "Exception: " << exp << endl;  }  try {  a7->setAt(2, 99);  a7->print();  }  catch (int exp) {  cout << "Exception: " << exp << endl;  }  delete a7; | 2 2 2 2 2 2 2 2 2 2  Exception: -1  Exception: -1  2 2 99 2 2 2 2 2 2 2 |

| Array a9(1000, 7);  a9.setAt(0, 99);  cout << "Subscript Array["  << 0 << "]:"  << a9[0] << endl;  a9[0] = -99;  cout << a9.getAt(0) << endl; | Subscript Array[0]:99  -99 |
| --- | --- |

| **Test** | **Result** |
| --- | --- |
| // Test copy constructor  Array \* a3;  Array \* a4 = new Array(10, 3);  a3 = new Array(\*a4);  delete a4;  a3->print();  delete a3; | 3 3 3 3 3 3 3 3 3 3 |