# Array of Objects

# Objectives

basic operations supported by an array.

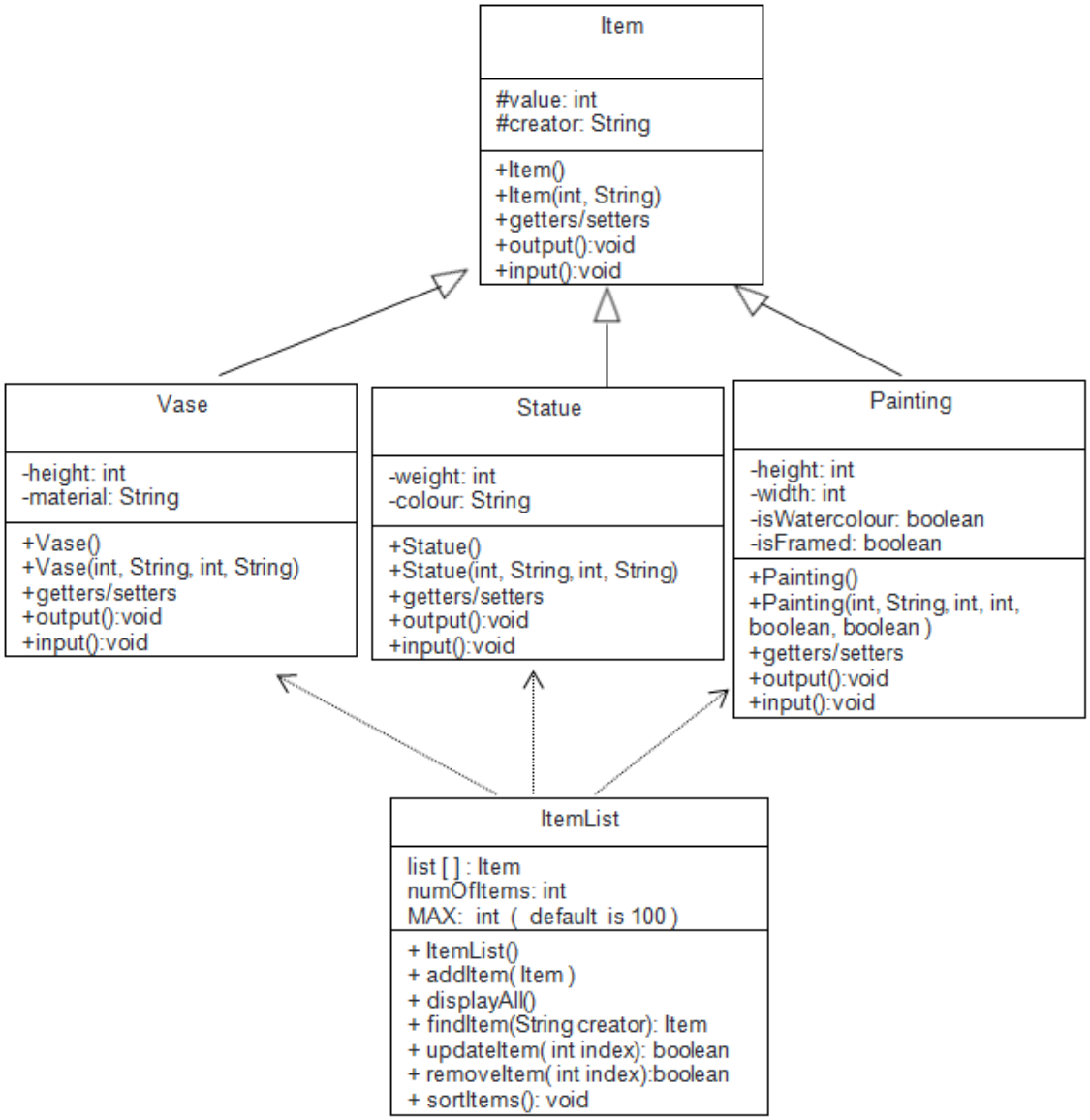
- print all the array elements(objects)
- Adds an object at the given index.
- Deletes an object at the given index.
- Searches an object using the given index or by the value.
- Updates an object at the given index.

# Case study

- **Problem**: A antique shop that sells antique items, namely vases, statues, and paintings. The owner can add item to inventory. The shop will keep items in the list. The owner can add a new item to it, he search also the item,….

=>For now, we want to manage the list of objects such as vases, statues, paintings in an array.

# Case study


Item
#value: int
#creator: String
+Item()
+Item(int, String)
+getters/setters
+output():void
+input():void
Vase
-height: int
-material: String
+Vase()
+Vase(int, String, int, String)
+getters/setters
+output():void
+input():void
Statue
-weight: int
-colour: String
+Statue()
+Statue(int, String, int, String)
+getters/setters
+output():void
+input():void
Painting
-height: int
-width: int
-isWatercolour: boolean
-isFramed: boolean
+Painting()
+Painting(int, String, int, int, boolean, boolean )
+getters/setters
+output():void
+input():void
ItemList
list [ ] : Item
numOfItems: int
MAX: int ( default is 100 )
+ ItemList()
+ addItem( Item )
+ displayAll()
+ findItem(String creator): Item
+ updateItem( int index): boolean
+ removeItem( int index):boolean
+ sortItems(): void

# Case study

```
public class Item
{
        protected int value;
         protected String creator;
        public Item(){      value=0;      creator="";    }
        public Item(int value, String creator){
              this.value=value;
              this.creator=creator;
        }
        //getter
        //setter
        public void input(){
           //use Scanner class to input fields
       }
       public void output(){
         //print fields out
       }
}
```

# Case study

```
public  class Vase extends Item
{
            private  int height;
            private String material;
            //constructors
            //getter
            //setter
            public void input(){
               //use Scanner class to input fields
            }
            public void output(){
              //print fields out
            }
}
```

```
public  class Statue extends Item
{
            private int weight;
            private  String colour;
            //constructors
            //getter
            //setter
            public void input(){
               //use Scanner class to input fields
            }
            public void output(){
              //print fields out
            }
}
```

# Case study

```
public  class  Painting extends Item
{
            private int height;
            private  int width;
            private  boolean isWatercolour;
            private  boolean isFramed;
            //constructors
            //getter
            //setter
            public void input(){
               //use Scanner class to input fields
            }
            public void output(){
              //print fields out
            }
}
```

# Case study

```
public  class  ItemList
{
        Item [] list;   // an array to store all items
        int numOfItem;  // to store the number of items that added to the list
        final int MAX=100;   // is the size of the array
        public ItemList(){
                list=new Item[MAX];
                numOfItem=0;
            }
        public boolean addItem(Item item){
                if( item==null || numOfItem>=MAX)
                   return false;
                list[numOfItem]=item;
                numOfItem++;
                return true;
      }
```

# case study

```
//this method prints out information of all items
public void displayAll(){
        if(numOfItem==0)
            System.out.println("the list is empty");
        for(int i=0; i< numOfItem; i++){
             System.out.println(list[i]);
     }
   }
   //this method finds the item by its creator
   //return the item that is found of the first occurrence.
   pulic Item findItem(String creator){
      for(int i=0; i< numOfItem; i++)
          if( list[i].getCreator().equals(creator))
              return list[i];
      return null;
   }
```

# Case study

```
//this method returns the zero_based index of the first occurrence.
    pulic int findItemIndex(String creator){
        for(int i=0; i< numOfItem; i++)
            if( list[i].getCreator().equals(creator))
                return i;
        return -1;
    }
    //this method updates the item at the specified position in this list
    //input: the index you wish to update

    pulic boolean updateItem(int index){
        if( index >= 0 && index < numOfItem){
             list[i].input();
             return true;
        }
        return false;
    }
```

# Case study

```
//this method removes the item at the specified position in this list.
    //Shifts any subsequent elements to the left
    //input: the index you wish to remove
    pulic boolean removeItem(int index){
        if( index >= 0 && index < numOfItem){
            for(int j=index; j< numOfItem; j++ ){
                list[j]=list[j+1];
             }
            numOfItem --;
            return true;
        }
        return false;
    }
```

# Case study

```
//this method prints out all items that belong to the given type in the list.
    public void displayItemsByType(String type){

        if (type.equals("Vase")){
          for(int i=0; i < numOfItem; i++)
             if ( list[i] instanceof Vase) System.out.println( list[i]);
        }
        else if (type.equals("Statue")){
          for(int i=0; i < numOfItem; i++)
             if ( list[i] instanceof Statue) System.out.println( list[i]);
        }
        else {
           for(int i=0; i < numOfItem; i++)
             if ( list[i] instanceof Painting) System.out.println( list[i]);
        }
    }
```

# Case study

```
//this method sorts items in ascending order based on their values.
   public void sortItems(){
       for(int i=0; i< numOfItem; i++)
           for(int j=numOfItem-1; j>i ;j--){
               if( list[i].getValue()< list[j-1].getValue()){
                   Item tmp=list[j];
                   list[j]=list[j-1];
                   list[j-1]=tmp;
                }
           }

   }

}//end class
```

# Case study

```
public class antiqueShop{
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int choice=0;
        do{
            System.out.println("1. add a new vase");
            System.out.println("2. add a new statue");
            System.out.println("3. add a new painting");
            System.out.println("4. display all items");
            System.out.println("5. find the items by the creator ");
            System.out.println("6. update the item by its index");
            System.out.println("7. remove the item by its index");
            System.out.println("8. display the list of vase items ");
            System.out.println("9. sorts items in ascending order based on their values ");
            System.out.println("10. exit");
            System.out.println("input your choice:");
            choice=sc.nextInt();
            switch(choice){
```

# Case study

```
            case 1:
                    Item tmp=new Vase();
                      tmp.input();
                      if(obj.addItem(tmp)){
                         System.out.println("added");
                      }
                      break;
                case 2:
                          …..
                      break
                case 3:
                          ….
                          break;
                 …..
        }//end switch
    } while(choice<=9);          //end while
}  //end class
```