

**Example:** Solidity program to demonstrate creating a fixed-size array

```
pragma solidity ^0.5.0;
// Creating a contract
contract Types {
    // Declaring state variables of type array
    uint[6] data1;
    // Defining function to add values to an array
    function array_example() public returns (
        int[5] memory, uint[6] memory){
        int[5] memory data= [int(50), -63, 77, -28, 90];
        data1 = [uint(10), 20, 30, 40, 50, 60];
        return (data, data1);
    }
}
```

**Example:** Demonstrate how to create and initialize dynamic arrays.

```
// creating a dynamic array
pragma solidity ^0.5.0;
// Creating a contract
contract Types {
    // Declaring state variable of type array. One is fixed-size and the other is dynamic array
    uint[] data = [10, 20, 30, 40, 50];
    int[] data1;
    // Defining function to assign values to dynamic array
    function dynamic_array() public returns(
        uint[] memory, int[] memory){
        data1 = [int(-60), 70, -80, 90, -100, -120, 140];
        return (data, data1);
    }
}
```

**Example: Accessing Array Elements:** the contract Types first initializes an *array[data]* and then retrieves the value at specific index 2.

```
// To demonstrate accessing elements of an array
pragma solidity ^0.5.0;
// Creating a contract
```

```

contract Types {
uint[6] data;
// Defining function to assign values to array
function array_example() public payable returns (uint[6] memory){
data = [uint(10), 20, 30, 40, 50, 60];
    return data;
}
// Defining function to access values from the array from a specific index
function array_element() public payable returns (uint){
    uint x = data[2];
    return x;
}
}

```

**Example: Length of Array:** the contract Types first initializes an *array[data]* and then the length of the array is calculated.

```

// Solidity program to demonstrate how to find length of an array
pragma solidity ^0.5.0;
// Creating a contract
contract Types {
uint[6] data;
// Defining a function to assign values to an array
    function array_example() public payable returns (uint[6] memory){
        data = [uint(10), 20, 30, 40, 50, 60];
        return data;
    }
// Defining a function to find the length of the array
function array_length() public returns(uint) {
    uint x = data.length;
    return x;
}
}

```

**Example: Push:** the contract Types first initializes an *array[data]*, and then more values are pushed into the array.

```

// Solidity program to demonstrate Push operation

```

```

pragma solidity ^0.5.0;
// Creating a contract
contract Types {
    // Defining the array
    uint[] data = [10, 20, 30, 40, 50];
    // Defining the function to push values to the array
    function array_push() public returns(uint[] memory){
        data.push(60);
        data.push(70);
        data.push(80);
        return data;
    }
}

```

**Example: Pop operation**

```

pragma solidity ^0.5.0;
contract Types {
// Creating a contract
uint[] data = [10, 20, 30, 40, 50]; // Defining an array
    // Defining a function to pop values from the array
    function array_pop() public returns(uint[] memory){
        data.pop();
        return data;
    }
}

```

**Example:** The *contract Types* consist of an *enumerator week\_days*, and functions are defined to set and get the value of a variable of the type enumerator.

```

// Solidity program to demonstrate how to use 'enumerator'
pragma solidity ^0.5.0;
contract Types {
enum week_days {Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday}
    // Declaring variables of type enumerator
    week_days week;
    week_days choice;
    // Setting a default value

```

```

week_days constant default_value = week_days.Sunday;
    // Defining a function to set value of choice
function set_value() public {
    choice = week_days.Thursday;
}
// Defining a function to return value of choice
function get_choice() public view returns (week_days) {
    return choice;
}
// Defining function to return default value
function getdefaultvalue() public pure returns(week_days) {
    return default_value;
}
}

```

**Example:** the *contract Test* consists of a structure *Book*, and functions are defined to set and get values of the elements of the structure.

```

// Solidity program to demonstrate how to use 'structures'
pragma solidity ^0.5.0;
// Creating a contract
contract test {
    // Declaring a structure
    struct Book {
        string name;
        string writer;
        uint id;
        bool available;
    }
    // Declaring a structure object
    Book book1;
    // Assigning values to the fields for the structure object book2
    Book book2 = Book("Building Ethereum DApps", "Roberto Infante ", 2, false);
    // Defining a function to set values for the fields for structure book1
    function set_book_detail() public {
        book1 = Book("Introducing Ethereum and Solidity", "Chris Dannen", 1, true);
    }
}

```

```
}  
// Defining function to print book2 details  
function book_info()public view returns (  
    string memory, string memory, uint, bool) {  
    return(book2.name, book2.writter,  
        book2.id, book2.available);  
}  
// Defining function to print book1 details  
function get_details() public view returns (string memory, uint) {  
    return (book1.name, book1.id);  
}  
}
```