

Be as proud of Sogang as Sogang is proud of you

블록체인 솔리디티 기초



서강대학교
SOGANG UNIVERSITY

- 가시성 지정자
- 조건문과 반복문
- Solidity 데이터 타입
 - 매핑(mapping)
 - 배열(Array)
 - 구조체(struct)
- 이벤트
- 생성자

■ 함수와 가시성 지정자

```
contract Ex4_21 {
    uint public pub = 1;
    uint private pri = 2;
    uint internal inter = 3;
    //uint external ext = 4;    // external은 변수 적용 불가

    function funPub() public view returns(uint,uint,uint){
        return (pub,pri,inter);
    }
    function funPriv() private view returns(uint,uint,uint){
        return (pub,pri,inter);
    }
    function funInter() internal view returns(uint,uint,uint){
        return (pub,pri,inter);
    }
    function funExt() external view returns(uint,uint,uint){
        return (pub,pri,inter);
    }
}
```

```
ParserError: Expected identifier
but got 'external'
--> week4/Ex4_21.sol:8:10:
|
8 | uint external ext = 4; //
   |          ^^^^^^^
external은 변수 적용 불가
```

EX4_21 AT 0XD7C...FAC5F (MEMORY)

Balance: 0 ETH

funExt

funPub

pub

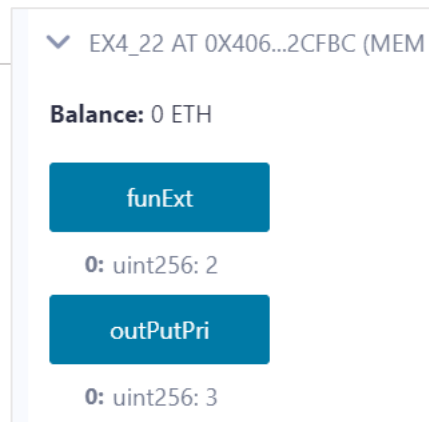
■ 함수와 가시성 지정자

- external 함수는 내부접근 불가

```
contract Ex4_22 {

    function funExt() external pure returns(uint) {
        return 2;
    }
    function funPri() private pure returns(uint) {
        return 3;
    }

    /*
    function outPutExt() public pure returns(uint) {
        return funExt();
    }
    */
    function outPutPri() public pure returns(uint) {
        return funPri();
    }
}
```



```
DeclarationError: Undeclared
identifier. "funExt" is not (or not
yet) visible at this point.
--> week3/Hello.sol:15:16:
|
15 | return funExt();
| ~~~~~
```

■ 함수와 가시성 지정자

- `this`를 이용해 external 함수 내부 접근

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.9.0;

contract Ex4_23 {

    function funExt() external pure returns(uint) {
        return 2;
    }

    function outPutExt() public view returns(uint) {
        return this.funExt();
    }

}
```

EX4_23 AT 0X081...A0F85 (MEMORY)

Balance: 0 ETH

funExt

outPutExt

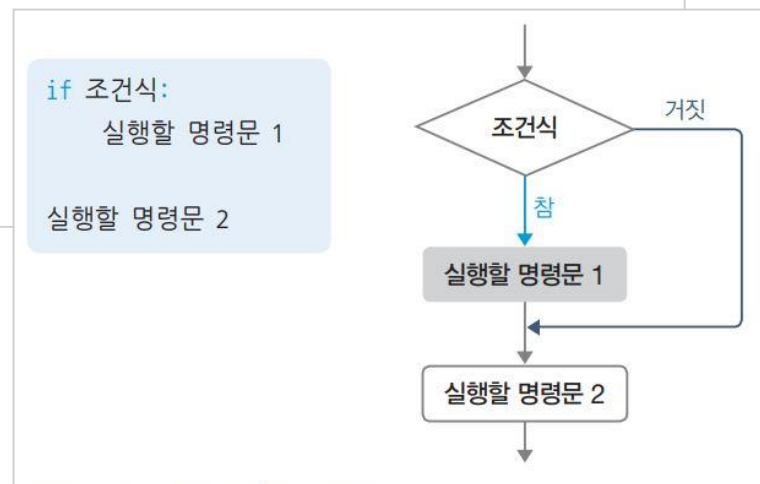
0: uint256: 2

■ 조건문(if)

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.9.0;

contract Ex4_24 {

    function fun1(uint a) public pure returns(uint) {
        if(a>=10){
            a = 9;
        }
        return a;
    }
}
```

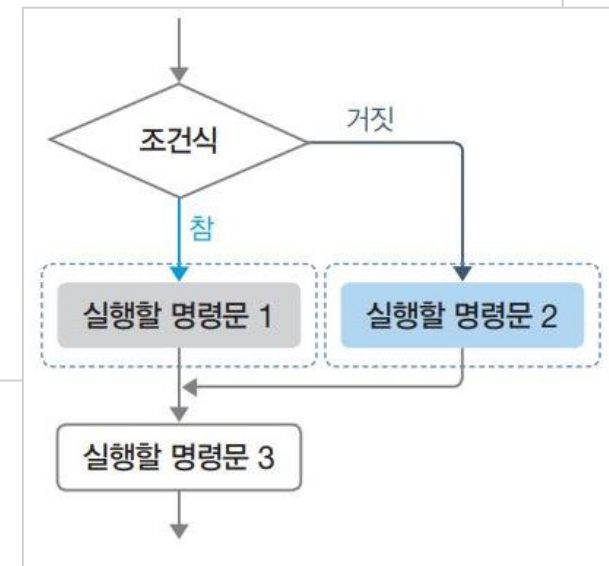


■ 조건문(if-else)

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.9.0;

contract Ex4_25 {

    function fun1(uint a) public pure returns(uint) {
        if(a>3){
            a = 9;
        }else{
            a = 10;
        }
        return a;
    }
}
```



■ 조건문(if-else if-else)

```
// SPDX-License-Identifier: GPL-3.0  
pragma solidity >=0.7.0 <0.9.0;
```

```
contract Ex4_26 {
```

```
    function fun1(uint a) public pure returns(uint) {
```

```
        if(a>=10){
```

```
            a = 9;
```

```
        }else if(a>=5&&a<=7){
```

```
            a = 7;
```

```
        }else{
```

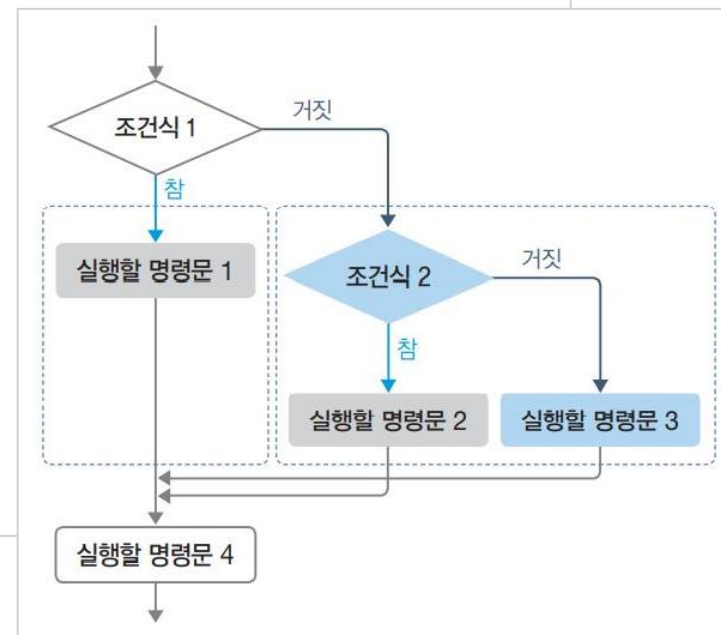
```
            a = 10;
```

```
        }
```

```
        return a;
```

```
    }
```

```
}
```



■ 조건문(if-else if-else): 잘못된 사용 예

```
contract Ex4_26_1 {
```

```
    function fun1(uint a) public pure returns(uint) {  
        if(a>=1){  
            a = 1;  
        }else if(a>=2){  
            a = 2;  
        }else if(a>=3){  
            a = 3;  
        }else{  
            a = 4;  
        }  
        return a;  
    }
```

```
    function fun2(uint a) public pure returns(uint) {  
        if(a>=1){  
            a = 1;  
        }  
        if(a>=2){  
            a = 2;  
        }  
        if(a>=3){  
            a = 3;  
        }else{  
            a = 4;  
        }  
        return a;  
    }  
}
```

■ 반복문의 필요성

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.9.0;

contract Ex4_27 {

    function fun1() public pure returns(uint) {
        uint a = 0;
        a += 1;
        a += 2;
        a += 3;
        a += 4;
        a += 5;
        a += 6;
        a += 7;
        a += 8;
        a += 9;
        a += 10;
        return a;
    }
}
```

■ for 반복문

```
for (초기식; 조건식; 증감식) {  
    // 반복문 로직  
}
```

```
// SPDX-License-Identifier: GPL-3.0  
pragma solidity >=0.7.0 <0.9.0;
```

```
contract Ex4_28 {  
  
    function fun1() public pure returns(uint) {  
        uint result = 0;  
        for(uint a = 0; a <3; ++a){  
            result = result + a;  
        }  
        return result;  
    }  
}
```

▼ EX4_28 AT 0XEC2_CF142 (MEMORY)

Balance: 0 ETH

fun1

0: uint256: 3

순서	a	result	a < 3
초기식	0		
조건식			true
반복문		0+0	
증감식	1		
조건식			true
반복문		0+1	
증감식	2		
조건식			true
반복문		1+2	
증감식	3		
조건식			false

■ while 반복문

```
초기식;
while (조건식) {

    // 반복문 로직

    증감식;
}
```

```
contract Ex4_29 {

    function fun1() public pure returns(uint) {
        uint result = 0;
        uint a = 3;
        while(a>0){
            result = result + a;
            --a;
        }

        return result;
    }
}
```

EX4_29 AT 0X939...78492 (MEMORY)

Balance: 0 ETH

fun1

0: uint256: 6

순서	a	result	a > 3
초기식	3		
조건식			true
반복문		0+3	
증감식	2		
조건식			true
반복문		3+2	
증감식	1		
조건식			true
반복문		5+1	
증감식	0		
조건식			false

■ do-while 반복문

```
초기식;
do {

    // 반복문 로직
    증감식;

} while (조건식)
```

```
contract Ex4_30 {

    function fun1() public pure returns(uint) {
        uint result = 0;
        uint a = 0;
        do{
            result = result + a;
            ++a;
        } while(a<3);

        return result;
    }
}
```

순서	a	result	a<3
초기식	0		
반복문		0+0	
증감식	1		
조건식			true
반복문		0+1	
증감식	2		
조건식			true
반복문		1+2	
증감식	3		
조건식			false

▼ EX4_30 AT 0XB57...9DF30 (MEMORY)

Balance: 0 ETH

fun1

0: uint256: 3

■ do-while 반복문: 최소한 한번은 실행

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.9.0;

contract Ex4_31 {

    function fun1() public pure returns(uint) {
        uint result = 5;
        uint a = 0;
        do {
            result = result + a;
            ++a;
        } while(a>10);

        return result;
    }
}
```

▼ EX4_31 AT 0X838...2A4DC (MEMORY)

Balance: 0 ETH

fun1

0: uint256: 5

■ 반복문과 조건문

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.9.0;

contract Ex4_32 {
    function fun1() public pure returns(uint) {
        uint result = 0;
        for(uint a = 0; a<10; ++a){
            if(a==1){
                return result;
            }
            result = result + a;
        }
        return result;
    }
}
```

▼ EX4_32 AT 0X9A2...BD189 (MEMORY)

Balance: 0 ETH

fun1

0: uint256: 0

■ 중첩반복문

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.9.0;

contract Ex4_33 {
    function fun1() public pure returns(uint) {
        uint result = 0;
        for(uint a = 1; a<2; ++a){
            for(uint b = 2; b<4; ++b){
                result = result + (a*b);
            }
        }
        return result;
    }
}
```

a	b	result
1	2	0+1*2
	3	2+1*3
	4	
2		

▼ EX4_33 AT 0X3C7...41288 (MEMORY)

Balance: 0 ETH

fun1

0: uint256: 5

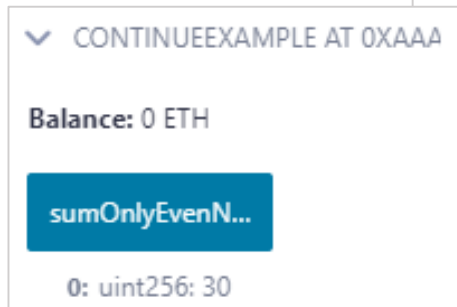
■ continue: 반복문의 맨 처음으로 돌아가기

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.9.0;

contract ContinueExample {

    function sumOnlyEvenNumbers() public pure returns(uint) {
        uint sumEven = 0; // 합계를 초기화

        for (uint i = 1; i <= 10; i++) {
            // 홀수는 건너뛰고 continue로 다음 반복으로 넘어감
            if (i % 2 != 0) {
                continue;
            }
            // 짝수만 합계에 더함
            sumEven += i;
        }
        return sumEven;
    }
}
```



■ break: 반복문 강제로 빠져나가기

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.9.0;

contract BreakInWhile {

    function findNumber() public pure returns(uint) {
        uint result;
        uint i = 1; // 초기화

        while (true) {
            // 특정 조건을 만족하면 반복문을 종료
            if (i == 5) {
                result = i; // 값을 result에 저장
                break;      // 반복문 종료
            }
            i++; // 증감식
        }
        return result;
    }
}
```

▼ BREAKINWHILE AT 0X0DB...8C5

Balance: 0 ETH

findNumber

0: uint256: 5

■ 매핑(mapping)

mapping	(address	=> uint)	public	balances
	키의자료형	값의자료형	가시성지정자	매핑명

- 키와 값 쌍으로 구성
- storage에 저장
- 상태변수로 선언
- 매핑은 길이를 지원하지 않음

■ 매핑(mapping)

- 매핑 데이터 저장과 조회: 키와 값 추가, 키로 값 얻기

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.9.0;

contract Ex5_1 {

    mapping(address => uint) public balances;

    function addMapping(address _key, uint _amount) public {
        balances[_key] = _amount;
    }


    function getMapping(address _key) public view returns(uint) {
        return balances[_key];
    }

}
```

■ 매핑(mapping)




- 매핑 데이터 저장과 조회: 키와 값 추가, 키로 값 얻기

DEPLOY & RUN TRANSACTIONS

ENVIRONMENT 

Remix VM (Cancun)

VM

ACCOUNT   

0x5B3...eddC4 (99.9999999999902)

GAS LIMIT

☒ Estimated Gas

☐ Custom 3000000


VALUE




0 Ether

CONTRACT

Ex5_1 - week5/Ex5_1.sol

계정주소복사

Deployed/Unpinned Contracts 



EX5_1 AT 0X7F1...D1F5D (MEMI)   

Balance: 0 ETH

addMapping

_key: 0x5B38Da6a701c568545dCfcB03Fcd


_amount: 1000




 Calldata  Parameters **transact**

balances address

getMapping address_key

키와 값 추가

Deployed/Unpinned Contracts 



EX5_1 AT 0X7F1...D1F5D (MEMI)   

Balance: 0 ETH

addMapping

_key: 0x5B38Da6a701c568545dCfcB03Fcd

_amount: 1000

 Calldata  Parameters **transact**

balances address

getMapping 0x5B38Da6a701c568545dCf

0: uint256: 1000

키로 값 얻기

■ 매핑(mapping)

■ 매핑 데이터 삭제: 키와 값 삭제

```
contract Ex5_2 {
```

```
    mapping(address => uint) public balances;
```

```
    function addMapping(address _key, uint _amount) public {
        balances[_key] = _amount;
    }
    function getMapping(address _key) public view returns(uint) {
        return balances[_key];
    }
}
```

```
    function deleteMapping1(address _key) public {
        delete(balances[_key]);
    }
```

```
    function deleteMapping2(address _key) public {
        balances[_key] = 0;
    }
}
```

Deployed Contracts

EX5_2 AT 0XD7A...F771B (MEMORY)

Balance: 0 ETH

addMapping

_key: 0x5B38Da6a701c568545dCfc803FcE

_amount: 1000

Calldata Parameters **transact**

deleteMapping1 address _key

deleteMapping2 address _key

balances 0x5B38Da6a701c568545dCf

0: uint256: 1000

getMapping address _key

Deployed Contracts

EX5_2 AT 0XD7A...F771B (MEMORY)

Balance: 0 ETH

addMapping

_key: 0x5B38Da6a701c568545dCfc803FcE

_amount: 1000

Calldata Parameters **transact**

deleteMapping1 0x5B38Da6a701c568545dCf

deleteMapping2 address _key

balances 0x5B38Da6a701c568545dCf

0: uint256: 0

getMapping 0x5B38Da6a701c568545dCf

0: uint256: 0

■ 배열(Array)

	92	108	76	14	99	55
index	0	1	2	3	4	5

uint	[]	public	arrayname
자료형		가시성지정자	배열이름

- 동일한 자료형을 순차적으로 저장
- 동적크기배열, 고정크기 배열 가능
- index: 0부터 시작, 순차적으로 증가(정수형)
- 배열의 길이 지원: 마지막 index + 1
- 배열의 각 요소는 **인덱스**를 통해 접근 가능

배열(Array)

- 배열의 길이 구하기, 인덱스에 대응하는 값 구하기

contract Ex5_3 {

```
uint[] public array1;
```

```
string[5] public array2 = ["apple", "banna", "Coconut"];
```

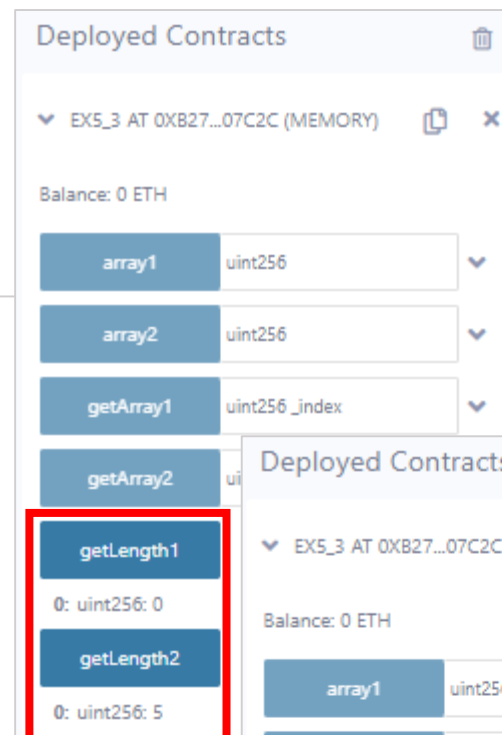
```
function getLength1() public view returns(uint) {  
    return array1.length;  
}
```

```
function getLength2() public view returns(uint) {  
    return array2.length;  
}
```

```
function getArray1(uint _index) public view returns(uint) {  
    return array1[_index];  
}
```

```
function getArray2(uint _index) public view returns(string memory)  
    return array2[_index];  
}
```

```
}
```

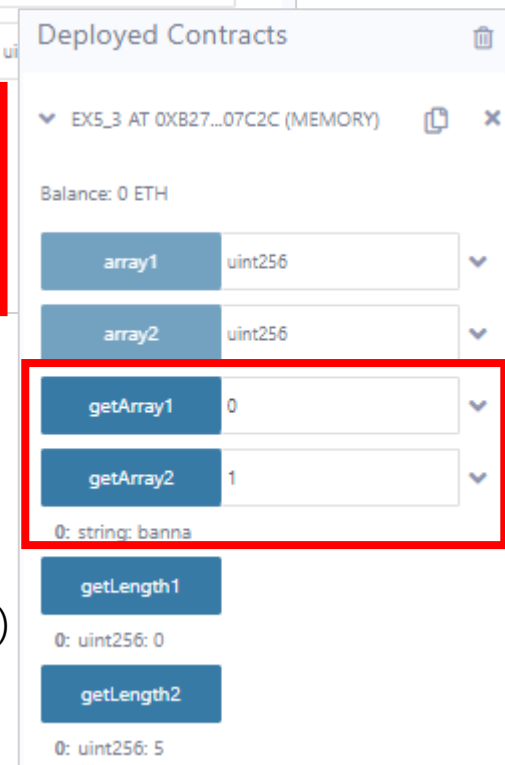


Deployed Contracts

EX5_3 AT 0XB27...07C2C (MEMORY)

Balance: 0 ETH

Function	Return Type
array1	uint256
array2	uint256
getArray1	uint256_index
getArray2	uint256_index
getLength1	uint256
getLength2	uint256



Deployed Contracts

EX5_3 AT 0XB27...07C2C (MEMORY)

Balance: 0 ETH

Function	Return Value
array1	uint256
array2	uint256
getArray1	0
getArray2	1
getLength1	0: uint256: 0
getLength2	0: uint256: 5

배열(Array)

- 배열의 길이 구하기, 인덱스에 대응하는 값 구하기
 - Index 범위밖 조회시 에러

getArray1 0

getArray2 5

0: string:

getLength1

```
CAL [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
L to: Ex5_3.getArray1(uint256) data: 0x29e...00000
call to Ex5_3.getArray1 errored: Error occurred: revert.

revert
The transaction has been reverted to the initial state.
Note: The called function should be payable if you send value and the v
Debug the transaction to get more information.
```

array2 uint256

getArray1 0

getArray2 5

0: string:

```
CAL [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
L to: Ex5_3.getArray2(uint256) data: 0x510...00005
call to Ex5_3.getArray2 errored: Error occurred: revert.

revert
The transaction has been reverted to the initial state.
Note: The called function should be payable if you send value and the v
Debug the transaction to get more information.
```

배열(Array)

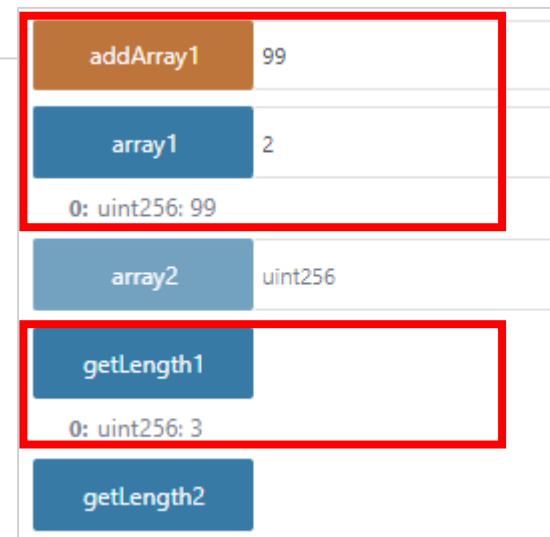
배열에 값 추가: push

```
contract Ex5_4 {
    uint[] public array1 = [97,98];
    string[5] public array2 = ["apple", "banana", "Coconut"];

    function getLength1() public view returns(uint) {
        return array1.length;
    }

    function getLength2() public view returns(uint) {
        return array2.length;
    }

    function addArray1(uint _value) public {
        array1.push(_value);
    }
    //오류 발생
    /*
    function addArray2(string memory _value) public {
        array2.push(_value);
    }
    */
}
```

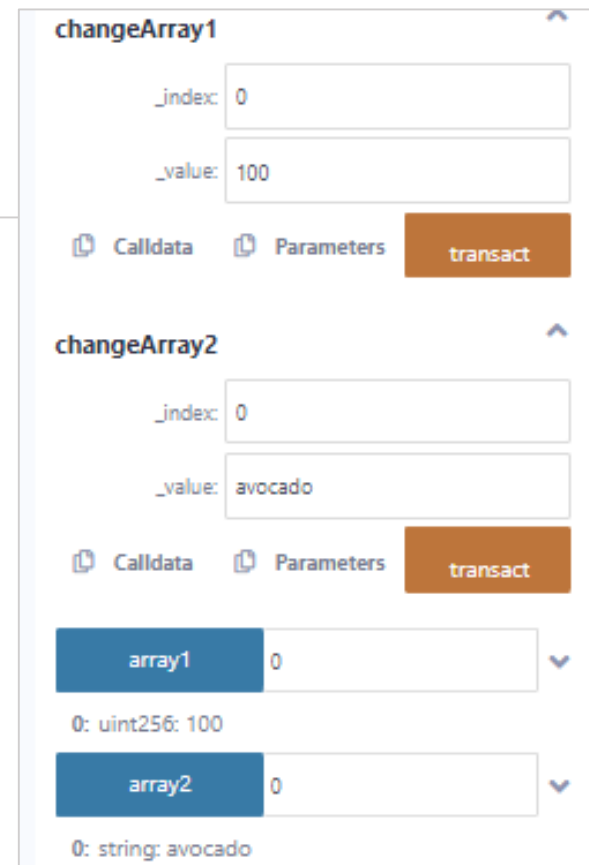


```
TypeError: Member "push" not found
or not visible after argument-
dependent lookup in string storage
ref[5] storage ref.
--> week3/Hello.sol:22:9:
|
|
22 | array2.push(_value);
|   ^^^^^^^^^^^
```

배열(Array)

배열 값 변경

```
contract Ex5_5 {  
  
    uint[] public array1 = [97,98];  
    string[5] public array2 = ["apple", "banana", "Coconut"];  
  
    function getLength1() public view returns(uint) {  
        return array1.length;  
    }  
    function getLength2() public view returns(uint) {  
        return array2.length;  
    }  
    function addArray1(uint _value) public {  
        array1.push(_value);  
    }  
    function changeArray1(uint _index, uint _value) public {  
        array1[_index] = _value;  
    }  
  
    function changeArray2(uint _index, string memory _value) public {  
        array2[_index] = _value;  
    }  
}
```



changeArray1

_index: 0

_value: 100

Calldata Parameters **transact**

changeArray2

_index: 0

_value: avocado

Calldata Parameters **transact**

array1 0

0: uint256: 100

array2 0

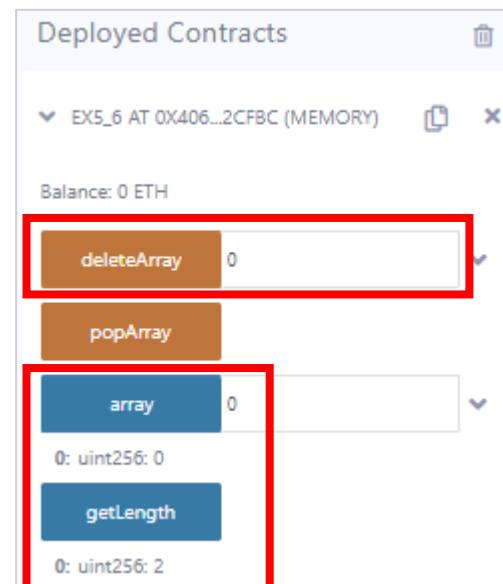
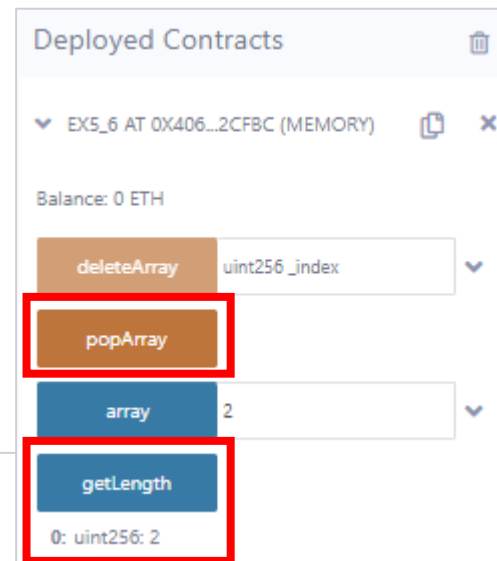
0: string: avocado

배열(Array)

배열의 값 삭제

- `pop()`: 마지막 원소 제거
- `delete`: 배열의 크기는 줄지 않는다

```
contract Ex5_6 {  
  
    uint[] public array = [97,98,99];  
  
    function getLength() public view returns(uint) {  
        return array.length;  
    }  
    function popArray() public {  
        array.pop();  
    }  
    function deleteArray(uint _index) public {  
        delete array[_index];  
    }  
}
```





배열(Array)


순차 검색 알고리즘

- 배열의 첫번째 인덱스부터 마지막 인덱스까지 순차적으로 검색
- 문자열 비교 기능이 없으므로 해시값 비교


```
contract Ex5_7 {  
  
    string[] public fruitArray = ["apple","banana","coconut","durian", "pear"];  
  
    function linearSearch(string memory _word) public view returns(uint256,string memory){  
  
        for(uint index = 0; index<fruitArray.length; ++index) {  
            if(keccak256(bytes(fruitArray[index])) == keccak256(bytes(_word))) {  
                return (index, fruitArray[index]);  
            }  
        }  
        return (0, "Nothing");  
    }  
}
```

EX5_7 AT 0X081...A0F85 (MEMORY)  


Balance: 0 ETH

fruitArray 3 

0: string: durian

linearSearch durian 

0: uint256: 3
1: string: durian

linearSearch mango 

0: uint256: 0
1: string: Nothing

■ 구조체(struct)

- 사용자 정의 자료형
- 한 개 이상 변수의 집단으로 구성
- 여러 다른 자료형을 결합하여 **복합적인 데이터**를 처리
- 구조체 정의

```
struct StructName {  
    자료형 필드1;  
    자료형 필드2;  
    자료형 필드3;  
}
```

```
struct Human {  
    string name;  
    uint age;  
    string job;  
}
```

```
Human public human1 = Human("ploy", 25, "artist");  
Human public human2;
```

```
function initializeHuman2(string memory _name, uint _age, string memory _job) public {  
    human2 = Human(_name, _age, _job);  
}
```

■ 구조체(struct)

■ 구조체 반환하는 함수

- 참조타입이므로 저장공간 명시



```
function getHuman1() public view returns(Human memory){  
    return human1;  
}  
function getHuman2() public view returns(Human memory){  
    return human2;  
}
```

■ 점(.) 연산자: 구조체 내부 변수 접근



```
function changeJobHuman1(string memory _job) public {  
    human1.job = _job;  
}  
function getNameHuman1() public view returns(string memory){  
    return human1.name;  
}
```



■ 구조체(struct)

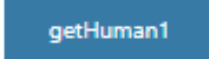
```
contract Ex5_8 {  
    struct Human {  
        string name;  
        uint age;  
        string job;  
    }  
  
    Human public human1 = Human("ploy", 25, "artist");  
    Human public human2;  
  
    function getHuman1() public view returns(Human memory){  
        return human1;  
    }  
  
    function getHuman2() public view returns(Human memory){  
        return human2;  
    }  
  
    function initializeHuman2(string memory _name, uint _age, string memory _job) public {  
        human2 = Human(_name, _age, _job);  
    }  
  
    function changeJobHuman1(string memory _job) public {  
        human1.job = _job;  
    }  
  
    function getNameHuman1() public view returns(string memory){  
        return human1.name;  
    }  
}
```

EX5_8 AT 0XE5F...78E22 (MEMORY)  

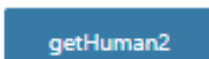
Balance: 0 ETH


 

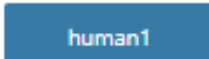


0: tuple(string,uint256,string): ploy,25,artist

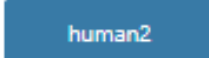


0: tuple(string,uint256,string): 0,





0: string: name ploy
1: uint256: age 25
2: string: job artist



0: string: name
1: uint256: age 0
2: string: job

■ 구조체(struct)

EX5_8 AT 0XE5F...78E22 (MEMORY)

Balance: 0 ETH

changeJobHu... string _job

initializeHuman2

_name: yujin
_age: 26
_job: doctor

Calldata Parameters **transact**

getHuman1
0: tuple(string,uint256,string): ploy,25,artist

getHuman2
0: tuple(string,uint256,string): yujin,26,doctor

getNameHum...

human1
0: string: name ploy
1: uint256: age 25
2: string: job artist

human2
0: string: name yujin
1: uint256: age 26
2: string: job doctor

EX5_8 AT 0XE5F...78E22 (MEMORY)

Balance: 0 ETH

changeJobHu... researcher

initializeHuman2 string _name, uint256 _age, :

getHuman1
0: tuple(string,uint256,string): ploy,25,researcher

getHuman2
0: tuple(string,uint256,string): yujin,26,doctor

getNameHum...
0: string: ploy

human1
0: string: name ploy
1: uint256: age 25
2: string: job artist

human2
0: string: name yujin
1: uint256: age 26
2: string: job doctor

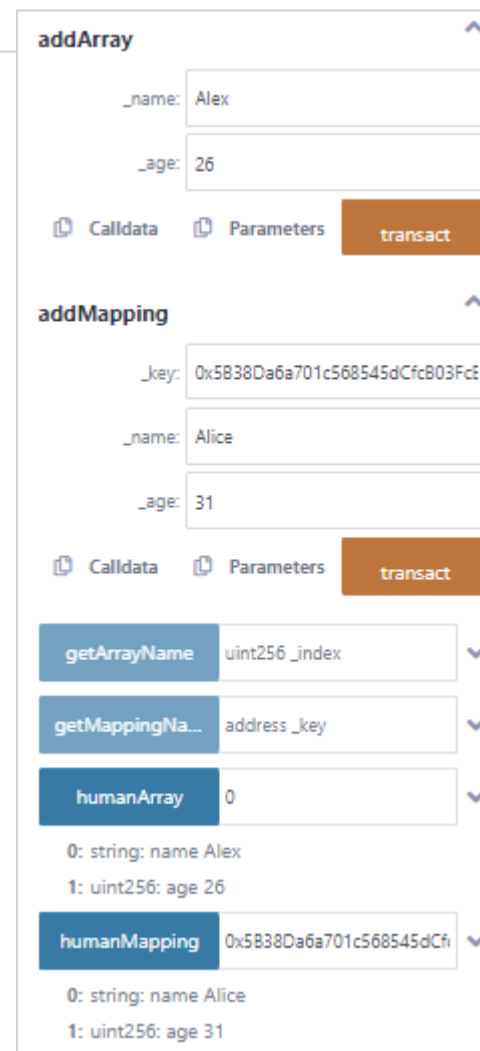
■ 구조체(struct)를 적용한 배열과 매핑

```
contract Ex5_9 {
    struct Human {
        string name;
        uint age;
    }

    Human[] public humanArray;
    mapping(address => Human) public humanMapping;

    function addArray(string memory _name, uint _age) public {
        humanArray.push(Human(_name, _age));
    }
    function getName(uint _index) public view returns(string memory) {
        return humanArray[_index].name;
    }

    function addMapping(address _key, string memory _name, uint _age) public {
        humanMapping[_key] = Human(_name, _age);
    }
    function getMappingName(address _key) public view returns(string memory) {
        return humanMapping[_key].name;
    }
}
```



addArray

_name: Alex

_age: 26

Calldata Parameters **transact**

addMapping

_key: 0x5838Da6a701c568545dCfc803FcE

_name: Alice

_age: 31

Calldata Parameters **transact**

getArrayName uint256 _index

getMappingName address _key

humanArray 0

0: string: name Alex
1: uint256: age 26

humanMapping 0x5838Da6a701c568545dCfc803FcE

0: string: name Alice
1: uint256: age 31

■ 이벤트

- 특정 데이터를 블록체인에 기록, 이후 검색 가능
- 스마트 컨트랙트 또는 유저의 특정한 상태를 출력
- Front-end 프로그램에 정보를 알리기 위해 사용
- indexed
 - event 매개변수에 사용할 수 있는 키워드
 - event를 검색 또는 필터링하는데 사용

■ 이벤트 정의와 출력

```
event MyEvent(uint result, string indexed name);
```

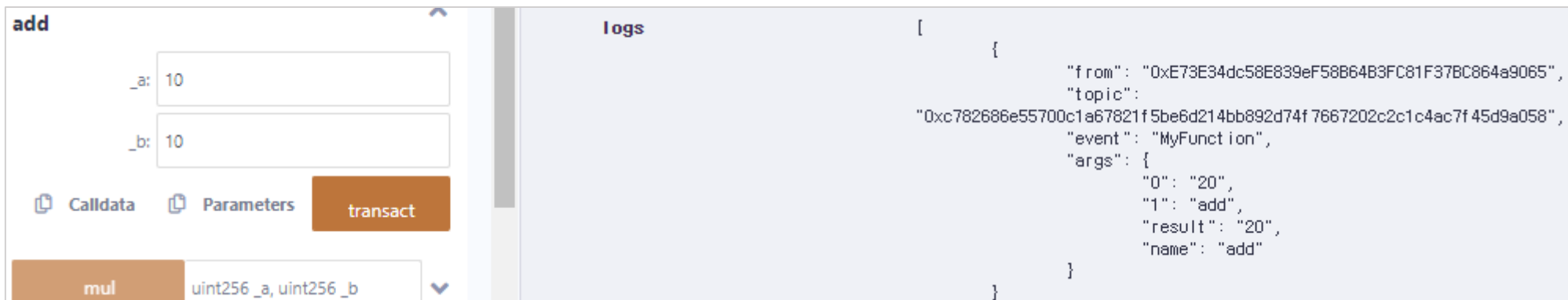
이벤트이름 이벤트 매개변수

```
emit MyEvent(10, "add");
```

이벤트이름

■ 이벤트

```
contract Ex5_10 {  
  
    event MyFunction(uint result, string name);  
  
    function add(uint _a, uint _b) public { // pure 나 view 불가  
        uint total = _a + _b;  
        emit MyFunction(total, "add");  
    }  
    function mul(uint _a, uint _b) public {  
        uint total = _a * _b;  
        emit MyFunction(total, "mul");  
    }  
}
```



The screenshot displays a Solidity IDE interface. On the left, the 'add' function is selected, showing input fields for '_a: 10' and '_b: 10'. Below these fields are buttons for 'Calldata', 'Parameters', and 'transact'. At the bottom, the 'mul' function is partially visible. On the right, the 'logs' panel shows a single log entry for the 'MyFunction' event. The log entry includes the transaction hash, the event name 'MyFunction', and the arguments: '0': '20', '1': 'add', 'result': '20', and 'name': 'add'.

```
[  
  {  
    "from": "0xE73E34dc58E839eF58B64B3FC81F37BC864a9065",  
    "topic":  
      "0xc782686e55700c1a67821f5be6d214bb892d74f7667202c2c1c4ac7f45d9a058",  
    "event": "MyFunction",  
    "args": {  
      "0": "20",  
      "1": "add",  
      "result": "20",  
      "name": "add"  
    }  
  }  
]
```

■ 생성자(constructor)

- 스마트 컨트랙트가 생성, 배포될 때 제일 먼저 한번 실행되는 함수
- 스마트 컨트랙트 배포시 변수에 특정한 값을 넣어줄 때 사용
- 생략 가능

```
// SPDX-License-Identifier: GPL-3.0  
pragma solidity >=0.7.0 <0.9.0;
```

```
contract Ex5_11 {  
    uint public num = 5;  
  
    constructor(uint _num) {  
        num = _num;  
    }  
}
```

CONTRACT

Ex5_11 - week3/Hello.sol

evm version: istanbul

Deploy uint256 _num

Deploy 55

☐ Publish to IPFS

At Address Load contract from Address

Transactions recorded 22

Deployed Contracts

EX5_11 AT 0X1C9...2B4BD (MEMORY)

Balance: 0 ETH

num

0: uint256: 55

■ immutable

- constant와 같이 한번 입력되면 변경 불가능
- constant는 선언시 초기값 필수, immutable은 필수 아님

```
contract Ex5_12 {  
  
    //uint public constant num1;  
    //uint[] public immutable arr;  
    uint public immutable num2;  
  
    constructor(uint _num) {  
        num2 = _num;  
    }  
    /* 생성자에서 초기화 가능  
    function change() public pure returns(uint) {  
        num2 = 10;  
    } */  
}
```

CONTRACT

Ex5_12 - week5.sol

evm version: istanbul

Deploy 100

☐ Publish to IPFS

At Address Load contract from Address

Transactions recorded 64 ⓘ ➤

Deployed Contracts ⓘ

EX5_12 AT 0X26A...92601 (MEMORY) ⓘ ✕

Balance: 0 ETH

num2

0: uint256: 100