1.

pragma solidity ^0.5.11;

contract Counter

{

int private count=0;

function incrementCounter() public {

count +=1;

}

function decrementCounter() public{

count -=1;

}

function getCount() public view returns (int)

{

return count;

}

}

2.

pragma solidity ^0.5.11;

contract SimpleStorage{

uint storeddata;

function set(uint x) public{

storeddata=x;

}

function get() public view returns(uint){

return storeddata;

}

}

3.

pragma solidity ^0.4.21;

contract yourToken{

address public minter;

mapping (address => uint) public balances;

event Sent(address from, address to, uint amount);

function yourToken() public{

minter = msg.sender;

}

function mint(address receiver , uint amount) public {

if(msg.sender!= minter) return;

balances[receiver]+= amount;

}

function send(address receiver, uint amount) public {

if(balances[msg.sender] < amount) return;

balances[msg.sender]-=amount;

balances[receiver]+=amount;

emit Sent(msg.sender, receiver, amount);

}

}

4.

pragma solidity ^0.5.11;

contract Mycontract

{

int private a=10;

int private b=20;

function adding() public view returns (int)

{

return a+b;

}

function subtract() public view returns (int)

{

return b-a;

}

function multiply() public view returns (int)

{

return b\*a;

}

function division() public view returns (int)

{

return b/a;

}

}

5.

pragma solidity ^0.5.11;

contract VRSE

{

string public str= "VRSE College";

}

6.

pragma solidity ^0.4.20;

contract Blocksplit {

address[] public players;

mapping (address => bool) public uniquePlayers;

address[] public winners;

address public charity = 0xc39eA9DB33F510407D2C77b06157c3Ae57247c2A;

uint256 drawnBlock = 0;

function() external payable {

play(msg.sender);

}

function play(address \_participant) payable public {

require (winners.length < 2);

require (msg.value >= 1000000000000000 && msg.value <= 100000000000000000);

require (uniquePlayers[\_participant] == false);

players.push(\_participant);

uniquePlayers[\_participant] = true;

}

function draw() external {

require (now > 1522908000);

require (block.number != drawnBlock);

require (winners.length < 2);

drawnBlock = block.number;

uint256 winningIndex = randomGen();

address winner = players[winningIndex];

winners.push(winner);

players[winningIndex] = players[players.length - 1];

players.length--;

if (winners.length == 2) {

charity.transfer(address(this).balance);

}

}

function randomGen() constant internal returns (uint256 randomNumber) {

uint256 seed = uint256(block.blockhash(block.number - 200));

return(uint256(keccak256(block.blockhash(block.number-1), seed )) % players.length);

}

}