**Solidity Assignment**

**1)Hello World using solidity**

//SPDX-License-Identifier:MIT

pragma solidity 0.8.10;

contract helloworld{

    string public str="Hello World";

}

**2)Program to find Factorial of number via pure functions**

//SPDX-License-Identifier:MIT

pragma solidity 0.8.10;

contract Factorial{

     function fact(uint x) public pure returns(uint){

         uint i;

         uint fac=1;

         for(i=1;i<=x;i++){

             fac=fac\*i;

         }

         return fac;

     }

}

**3)Implementing decentralised voting system for 3 candidates, each voter can vote twice**

**4)Palindrome Program : to write a code to return palindrome of a string, if it is palindrome transfer 50 ETH from one account to manager account.**

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.10;

contract Palindrome{

    address payable Manager = payable(0xfEABA578D94C682c04BBE757f560A87eD8e4C7Aa);

    function payEther() public payable{

    }

    function sendEther(string memory word) public returns(string memory){

        bytes memory new\_word = bytes(word);

        string memory len = new string(new\_word.length);

        bytes memory nw1 = bytes(len);

        for(uint i = 0; i<new\_word.length; i++){

            nw1[new\_word.length-i-1] = new\_word[i];

        }

        string memory s;

        if(keccak256(abi.encodePacked(word)) == keccak256(abi.encodePacked(nw1))){

            Manager.transfer(50 ether);

            s = "SendingDone";

        }

        else{

            s = "false";

        }

        return s;

    }

}

**5)Write a contract 'Time' which implements a function named getTime.**

**6)Problem Statement**

//SPDX-License-Identifier:MIT

pragma solidity ^0.8.10;

contract ThreeAndSeven{

    function check(uint a) public pure returns(bool){

        if((a%3==0 || a%7==0) && a>10){

            return true;

        }

        else{

            return false;

        }

    }

}

**7)EvenOdd**

//SPDX-License-Identifier:MIT

pragma solidity ^0.8.10;

contract Evenodd{

    function check(uint a) public pure returns(string memory){

        string memory i;

        if(a%2==0){

            i="Even Number";

            return i;

        }

        else{

            i="Odd Number";

            return i;

        }

    }

}

**8)Problem Statement**

//SPDX-License-Identifier:MIT

pragma solidity ^0.8.10;

contract Calculator{

    function add(uint a,uint b) public pure

    returns(uint){

        uint x;

        x=a+b;

        return x;

    }

    function subtract(uint a,uint b) public pure

    returns(uint){

        uint x;

        x=a-b;

        return x;

    }

    function divide(uint a,uint b) public pure

    returns(uint){

        uint x;

        x=a/b;

        return x;

    }

    function multiply(uint a,uint b) public pure

    returns(uint){

        uint x;

        x=a\*b;

        return x;

    }

    function power(uint a) public pure

    returns(uint){

        uint x;

        x=a\*a;

        return x;

    }

    function returnAll(uint a, uint b) public pure returns (uint,uint,uint,uint){

return (a\*\*b,a+b,a\*b,a/b);

}

}

**9)Program to find Sum of Digits**

//SPDX-License-Identifier:MIT

pragma solidity ^0.8.10;

contract sum{

    function digit(uint a, uint b) public pure returns(uint){

        uint c;

        c=a+b;

        return c;

    }

}

**10)Problem Statement**

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.10;

contract AttendanceRegister{

    struct student{

        string Name;

        uint Class;

        string JoiningDate;

        }

        address public Teacher = msg.sender;

        mapping (uint => student) public data;

        event Register(address Teacher, student Data);

        modifier onlyTeacher() {

        require(Teacher == msg.sender, "You are not a teacher");

        \_;

        }

        function add(uint check, string memory name, uint class, string memory date) public onlyTeacher {

            data[check] = student(name,class,date);

            emit Register(msg.sender, data[check]);

            }

            }