pragma solidity >=0.4.0<=0.6.0;

contract StudentRegister{

    address public owner;

    mapping (address=>student)students;

    constructor() public {

        owner=msg.sender;

    }

    /\*\*

     \*a modifier onlyOwner is created to limit the access to function register to contract deployer

     \*/

    modifier onlyOwner {

        require(msg.sender==owner);

        \_;

    }

    /\*\*

     \*a struct student is defined

     \*/

    struct student{

        address studentId;

        string  name;

        string course;

        uint256 mark1;

        uint256 mark2;

        uint256 mark3;

        uint256 totalMarks;

        uint256 percentage;

        bool isExist;

    }

    function register(address studentId,string memory name,string memory course,uint256 mark1,uint256 mark2,uint256 mark3) public onlyOwner {

            require(students[studentId].isExist==false,"ha.. ha... Fraud Not Possible,student details already registered and cannot be altered");

            uint256 totalMarks;

            uint256 percentage;

            /\*\*

             \*calculating totalMarks and percentage

             \*/

            totalMarks=(mark1+mark2+mark3);

            percentage=(totalMarks/3);

            /\*\*

             \*  assigning the student details to a key (studentId)

             \*/

            students[studentId]=student(studentId,name,course,mark1,mark2,mark3,totalMarks,percentage,true);

    }

    function getStudentDetails(address studentId) public view returns (address,string memory,string memory,uint256,uint256){

        /\*\*

         \*returning studentId,name,course,totalMarks and percentage of student to corresponding key

         \*/

        return(students[studentId].studentId,students[studentId].name,students[studentId].course,students[studentId].totalMarks,students[studentId].percentage);

    }

}