

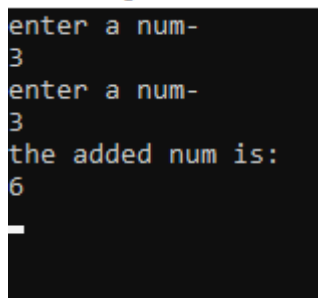
# Aug-3

## C#

### PROGRAM TO ADD TWO NUMBERS

```
using System;

namespace simple
{
    class Program
    {
        static void Main(string[] args)
        {
            double num1, num2;
            int res;
            Console.WriteLine("enter a num-");
            num1 = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("enter a num-");
            num2 = Convert.ToInt32(Console.ReadLine());
            res = (int) (num1 + num2);
            Console.WriteLine("the added num is:");
            Console.WriteLine(res);
            Console.ReadLine();
        }
    }
}
```



```
enter a num-
3
enter a num-
3
the added num is:
6
```

### PROGRAM TO FIND THE ARMSTRONG NUMBER

```
static void Main(string[] args)
{
    int num, cub=0, sum = 0;
    Console.WriteLine("enter a num");
    num = Convert.ToInt32(Console.ReadLine());
    int temp = num;
    while (num>0)
    {
        int rem = num % 10;
        cub+= (rem*rem*rem);
        sum += rem;
        num /= 10;
    }

    if (temp==cub)
    {
        Console.WriteLine("ARMSTRONG NUMBER");
        Console.ReadLine();
    }
}
```

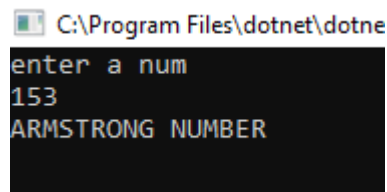
```

    }
    else
    {
        Console.WriteLine("not an armstrong number");
        Console.ReadLine();
    }
}

}

}

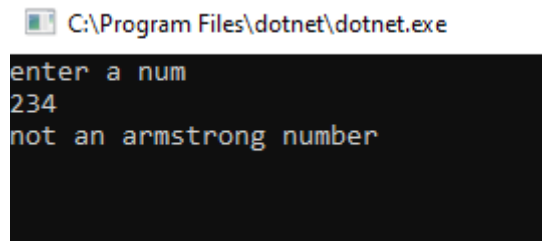
```



```

C:\Program Files\dotnet\dotnet
enter a num
153
ARMSTRONG NUMBER

```

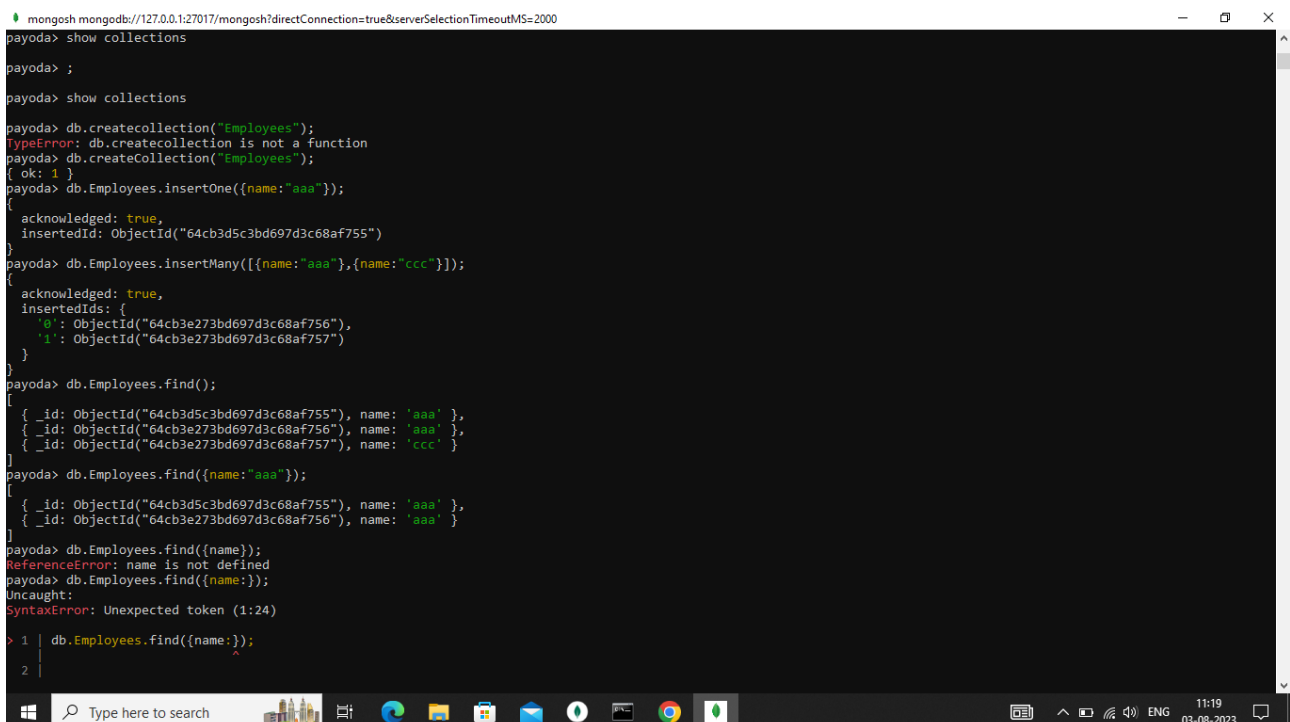


```

C:\Program Files\dotnet\dotnet.exe
enter a num
234
not an armstrong number

```

## MONGODB



```

mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000
payoda> show collections
payoda> ;
payoda> show collections
payoda> db.createcollection("Employees");
TypeError: db.createcollection is not a function
payoda> db.createCollection('Employees');
{ ok: 1 }
payoda> db.Employees.insertOne({name:"aaa"});
{ acknowledged: true,
  insertedId: ObjectId("64cb3d5c3bd697d3c68af755") }
payoda> db.Employees.insertMany([{name:"aaa"},{name:"ccc"}]);
{ acknowledged: true,
  insertedIds: {
    '0': ObjectId("64cb3e273bd697d3c68af756"),
    '1': ObjectId("64cb3e273bd697d3c68af757")
  } }
payoda> db.Employees.find();
[ { _id: ObjectId("64cb3d5c3bd697d3c68af755"), name: 'aaa' },
  { _id: ObjectId("64cb3e273bd697d3c68af756"), name: 'aaa' },
  { _id: ObjectId("64cb3e273bd697d3c68af757"), name: 'ccc' } ]
payoda> db.Employees.find({name:"aaa"});
[ { _id: ObjectId("64cb3d5c3bd697d3c68af755"), name: 'aaa' },
  { _id: ObjectId("64cb3e273bd697d3c68af756"), name: 'aaa' } ]
payoda> db.Employees.find({name});
ReferenceError: name is not defined
payoda> db.Employees.find({name:});
Uncaught:
SyntaxError: Unexpected token (1:24)
> 1 | db.Employees.find({name:});
  2 |

```

- creating a collection(Employee) the db (payoda)



```
mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000
{ "_id": ObjectId("64cb3e273bd697d3c68af757"), name: 'ccc' },
{ "_id": ObjectId("64cb409c3bd697d3c68af758"), city: 'cbe' },
{ "_id": ObjectId("64cb49c99e66b3a0f86d9c55"), name: 'xyz' },
{ "_id": ObjectId("64cb4a1a9e66b3a0f86d9c88"), name: 'abc' }
]
payoda> db.Employees.updateMany({name:"xyz"},{$set:{name:"abc"}},{upsert:true})
{ acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0 }
payoda> db.Employees.find();
[
  { "_id": ObjectId("64cb3d5c3bd697d3c68af755"), name: 'abc' },
  { "_id": ObjectId("64cb3e273bd697d3c68af756"), name: 'aaa' },
  { "_id": ObjectId("64cb3e273bd697d3c68af757"), name: 'ccc' },
  { "_id": ObjectId("64cb409c3bd697d3c68af758"), city: 'cbe' },
  { "_id": ObjectId("64cb49c99e66b3a0f86d9c55"), name: 'abc' },
  { "_id": ObjectId("64cb4a1a9e66b3a0f86d9c88"), name: 'abc' }
]
payoda> db.Employees.deleteOne({name:"xyz"})
{ acknowledged: true, deletedCount: 0 }
payoda> db.Employees.find();
[
  { "_id": ObjectId("64cb3d5c3bd697d3c68af755"), name: 'abc' },
  { "_id": ObjectId("64cb3e273bd697d3c68af756"), name: 'aaa' },
  { "_id": ObjectId("64cb3e273bd697d3c68af757"), name: 'ccc' },
  { "_id": ObjectId("64cb409c3bd697d3c68af758"), city: 'cbe' },
  { "_id": ObjectId("64cb49c99e66b3a0f86d9c55"), name: 'abc' },
  { "_id": ObjectId("64cb4a1a9e66b3a0f86d9c88"), name: 'abc' }
]
payoda> db.Employees.deleteMany({name:"abc"})
{ acknowledged: true, deletedCount: 3 }
payoda> db.Employees.find();
[
  { "_id": ObjectId("64cb3e273bd697d3c68af756"), name: 'aaa' },
  { "_id": ObjectId("64cb3e273bd697d3c68af757"), name: 'ccc' },
  { "_id": ObjectId("64cb409c3bd697d3c68af758"), city: 'cbe' }
]
payoda> db.Employees.deleteMany({})
MongoshInvalidInputError: [COMMON-10001] Missing required argument at position 0 (Collection.deleteMany)
payoda>
```

updating the collection in database using

- db.employees.updatemany()

and delete the data in collection using

- db.employees.deleteOne()
- db.employees.deleteMany()

```
mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000
insertedIds: {
  '0': ObjectId("64cb4be63bd697d3c68af759"),
  '1': ObjectId("64cb4be63bd697d3c68af75a")
}
payoda> db.Employees.deleteOne({})
{ acknowledged: true, deletedCount: 1 }
payoda> db.Employees.find()
[
  {
    _id: ObjectId("64cb4be63bd697d3c68af75a"),
    name: 'xxx',
    city: 'cbe'
  }
]
payoda> db.Employees.insertOne({name:"ppp",city:"cbe"});
{ acknowledged: true,
  insertedId: ObjectId("64cb4c1a3bd697d3c68af75b") }
payoda> db.Employees.find()
[
  {
    _id: ObjectId("64cb4be63bd697d3c68af75a"),
    name: 'xxx',
    city: 'cbe'
  },
  {
    _id: ObjectId("64cb4c1a3bd697d3c68af75b"),
    name: 'ppp',
    city: 'cbe'
  }
]
payoda> db.Employees.deleteOne({})
{ acknowledged: true, deletedCount: 1 }
payoda> db.Employees.find()
[
  {
    _id: ObjectId("64cb4c1a3bd697d3c68af75b"),
    name: 'ppp',
    city: 'cbe'
  }
]
payoda>
```

The data are deleted and the acknowledgement is given.

```
mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000
payoda> db.users.find({age:"22"},{name:1,city:1,_id:1});
[
  {
    _id: ObjectId("64cb4d273bd697d3c68af75c"),
    name: 'preethi',
    city: 'cbe'
  }
]
payoda> db.users.find({age:"22"},{name:1,city:1,_id:1,age:1});
[
  {
    _id: ObjectId("64cb4d273bd697d3c68af75c"),
    name: 'preethi',
    city: 'cbe',
    age: '22'
  }
]
payoda> db.users.find({age:"22"},{name:0,city:0,_id:1,age:0});
[ { _id: ObjectId("64cb4d273bd697d3c68af75c") } ]
payoda> db.users.find({age:"22"},{name:0,city:0,_id:1,age:1});
MongoServerError: Cannot do inclusion on field age in exclusion projection
payoda> db.users.insertOne({name:"rithika",city:"chennai",age:"21",});
{
  acknowledged: true,
  insertedId: ObjectId("64cb4f893bd697d3c68af75d")
}
payoda> db.users.find( {name:"rithika",age:{$gt:21}});
payoda> db.users.find( {name:"rithika",age:{$gt:20}});
payoda> db.users.find( {name:"rithika",age:{$gt:20}}, {name:1});
payoda> db.users.find( {name:"rithika",age:{$gt:20}}, {name:1});
[ { _id: ObjectId("64cb4f893bd697d3c68af75d"), name: 'rithika' } ]
payoda> db.users.find( {name:"rithika",age:{$lt:22}}, {name:1});
[ { _id: ObjectId("64cb4f893bd697d3c68af75d"), name: 'rithika' } ]
payoda> db.users.find( {name:"rithika",age:{$lt:22}}, {name:1});
[ { _id: ObjectId("64cb4f893bd697d3c68af75d"), name: 'rithika' } ]
payoda> db.users.find( {name:"rithika",age:{$lt:22}}, {name:1});
[ { _id: ObjectId("64cb4f893bd697d3c68af75d"), name: 'rithika' } ]
payoda> db.users.find( {age:{$lt:25}}, {name:1});
[
  {
    _id: ObjectId("64cb4d273bd697d3c68af75c"), name: 'preethi' },
  {
    _id: ObjectId("64cb4f893bd697d3c68af75d"), name: 'rithika' }
]
payoda>
```

Creating a new db (users)

- db use users

Finding the data in the collection

- db.users.find()

```
mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000
SyntaxError: Unexpected token, expected ",", (1:32)
> 1 | db.users.find( {age:{$in:['20','25']}}, {name:1});
  |               ^
  |
  |
payoda> db.users.find( {age:{$lt:'30',$gt:'20'}}, {name:1});
Uncaught:
SyntaxError: Unexpected token, expected ",", (1:34)
> 1 | db.users.find( {age:{$lt:'30',$gt:'20'}}, {name:1});
  |               ^
  |
  |
payoda> db.users.find( {age:{$lt:'30',$gt:'20'}}, {name:1});
[
  {
    _id: ObjectId("64cb4d273bd697d3c68af75c"), name: 'preethi' },
  {
    _id: ObjectId("64cb4f893bd697d3c68af75d"), name: 'rithika' }
]
payoda> db.users.find( {age:{$lt:'30',$gt:'20'}}, {name:1});
payoda> db.users.aggregate([{}])
Uncaught:
SyntaxError: Unexpected token (1:21)
> 1 | db.users.aggregate([{}])
  |               ^
  |
  |
payoda> db.users.aggregate([{$match:{name:preethi}},{$count:name}])
ReferenceError: preethi is not defined
payoda> db.users.aggregate([{$match:{name:"preethi"}},{$count:name}])
ReferenceError: name is not defined
payoda> db.users.aggregate([{$match:{name:"preethi"}},{$count}])
ReferenceError: $count is not defined
payoda> db.users.aggregate([{$match:{name:"preethi"}},{$count:}])
Uncaught:
SyntaxError: Unexpected token (1:54)
> 1 | db.users.aggregate([{$match:{name:"preethi"}},{$count:}])
  |               ^
  |
  |
payoda> db.users.aggregate([{$match:{name:"preethi"}},{$count:}])
```

aggregation function is give using

- db.users.aggregate()

```
mongosh mongodb://127.0.0.1:27017/mongosh?directConnections=true&serverSelectionTimeoutMS=2000
{
  _id: ObjectId("64cb4d273bd697d3c68af75c"),
  name: 'preethi',
  city: 'cbe'
}
payoda> db.users.find({age:"22"},{name:1,city:1,_id:1});
[
  {
    _id: ObjectId("64cb4d273bd697d3c68af75c"),
    name: 'preethi',
    city: 'cbe'
  }
]
payoda> db.users.find({age:"22"},{name:1,city:1,_id:1,age:1});
[
  {
    _id: ObjectId("64cb4d273bd697d3c68af75c"),
    name: 'preethi',
    city: 'cbe',
    age: '22'
  }
]
payoda> db.users.find({age:"22"},{name:0,city:0,_id:1,age:0});
[ { _id: ObjectId("64cb4d273bd697d3c68af75c") } ]
payoda> db.users.find({age:"22"},{name:0,city:0,_id:1,age:1});
MongoServerError: Cannot do inclusion on field age in exclusion projection
payoda> db.users.insertOne({name:"rithika",city:"chennai",age:"21",})
{
  acknowledged: true,
  insertedId: ObjectId("64cb4f893bd697d3c68af75d")
}
payoda> db.users.find( {name:"rithika",age:{$gt:21}});
payoda> db.users.find( {name:"rithika",age:{$gt:20}});
payoda> db.users.find( {name:"rithika",age:{$gt:20}}, {name:1});
payoda> db.users.find( {name:"rithika",age:{$gt:'20'}},{name:1});
[ { _id: ObjectId("64cb4f893bd697d3c68af75d"), name: 'rithika' } ]
payoda> db.users.find( {name:"rithika",age:{$lt:'22'}},{name:1});
[ { _id: ObjectId("64cb4f893bd697d3c68af75d"), name: 'rithika' } ]
payoda> db.users.find( {name:"rithika",age:{$lt:'25'}},{name:1});
[ { _id: ObjectId("64cb4f893bd697d3c68af75d"), name: 'rithika' } ]
payoda> db.users.find( {age:{$lt:'25'}},{name:1});
```

using db.users.find with (AND ,OR)condition