Ranganathan.kr **Bootstrap assignment** **3.8.2023**

Task:

1. **Make use of the queries given by the trainer in your mongoDB.**
2. **Write C# code to check for perfect number.**
3. **Write C# code to generate prime numbers in given range.**

1.Queries:

```
MongoshInvalidInputError: [COMMON-10001] 'local' is not a valid argument for "show".
testbae> show testbae;
MongoshInvalidInputError: [COMMON-10001] 'testbae' is not a valid argument for
"show".
testbae> db testbae;
Uncaught:
SyntaxError: Missing semicolon. (1:2)

> 1 | db testbae;
    |   ^
  2 |

testbae> use testbae;
already on db testbae
testbae> use testbae
already on db testbae
testbae> db;
testbae
testbae> use testbae
already on db testbae
testbae> db.createCollection("details");
MongoServerError: Collection testbae.details already exists.
testbae> db.details.insertOne({name:"ice cream"})
{
  acknowledged: true,
  insertedId: ObjectId("64cb3d4df5bc638d2760131c")
}
testbae>
db.details.insertMany([{name:"noodles",add:"jupiter"},{name:"shell",add:"neptune"}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("64cb3e28f5bc638d2760131d"),
    '1': ObjectId("64cb3e28f5bc638d2760131e")
  }
```

```
}
testbae> db.details.find()
[
  { _id: 100, name: 'armster', Add: 'Mars' },
  {
    _id: ObjectId("64cb3d4df5bc638d2760131c"),
    name: 'ice cream',
    add: 'mercury'
  },
  {
    _id: ObjectId("64cb3e28f5bc638d2760131d"),
    name: 'noodles',
    add: 'jupiter'
  },
testbae> db.details.find({name:"shell"},{name:1,add:1})
[   _id: ObjectId("64cb3e28f5bc638d2760131e"),
  { name: 'shell',
    _id: ObjectId("64cb3e28f5bc638d2760131e"),
    name: 'shell',
    add: 'neptune'
  }tbae> db.details.find({name:"shell"})
]
testbae> db.details.find({name:"shell"},{name:1,add:1})
[   _id: ObjectId("64cb3e28f5bc638d2760131e"),
  { name: 'shell',
    _id: ObjectId("64cb3e28f5bc638d2760131e"),
    name: 'shell',
    add: 'neptune'
  }tbae> db.details.find([{name:"shell"},{name:"armster"}])
]ongoInvalidArgumentError: Query filter must be a plain object or ObjectId
testbae> db.details.find({name:"shell"},{name:0,add:0})
[ { _id: ObjectId("64cb3e28f5bc638d2760131e") } ]d in exclusion projection
testbae> db.details.update({name:"noodles"},{$set{add:"venus"}})
Uncaught:ObjectId("64cb3e28f5bc638d2760131e"), name: 'shell' } ]
SyntaxError: Unexpected token, expected "," (1:40)
[
> 1 | db.details.update({name:"noodles"},{$set{add:"venus"}})
    |                                        ^
  2 |id: ObjectId("64cb3d4df5bc638d2760131c"),
    name: 'ice cream',
testbae> db.details.update({name:"noodles"},{$set: {add:"venus"}})
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or
bulkWrite.
{ {
  acknowledged: true,cb3e28f5bc638d2760131d"),
  insertedId: null,,
  matchedCount: 1,
  modifiedCount: 1,
```

```
  upsertedCount: 0
}   _id: ObjectId("64cb3e28f5bc638d2760131e"),
testbae> db.details.find()
[   add: 'neptune'
  { _id: 100, name: 'armster', Add: 'Mars' },
  {
    _id: ObjectId("64cb3d4df5bc638d2760131c"),
    name: 'ice cream',ot do exclusion on field add in inclusion projection
    add: 'mercury'
  },
  {
    _id: ObjectId("64cb3e28f5bc638d2760131d"),
    name: 'noodles',
    add: 'venus'
  },
  {
    _id: ObjectId("64cb3e28f5bc638d2760131e"),
    name: 'shell',
    add: 'neptune'
  }
]
testbae> db.details.delete({name:"shell"})
TypeError: db.details.delete is not a function
testbae> db.details.deleteOne({name:"shell"})
{ acknowledged: true, deletedCount: 1 }
testbae> db.details.deleteOne({})
{ acknowledged: true, deletedCount: 1 }
testbae> db.details.deleteMany({})
{ acknowledged: true, deletedCount: 2 }
testbae> db.details.find()

testbae> db.details.insert({name:"armster",add:"pluto"})
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or
bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64cb4b45f5bc638d2760131f") }
}
testbae> db.details.find()
[
  {
    _id: ObjectId("64cb4b45f5bc638d2760131f"),
    name: 'armster',
    add: 'pluto'
  }
]
testbae> db.details.updateOne({name:"armster"},{$rat :{add:"7"}})
```

```
MongoServerError: Unknown modifier: $rat. Expected a valid update modifier or pipeline-style update specified as an array
testbae> db.details.updateOne({name:"armster"},{$set :{rat:"7"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
testbae> db.details.updateOne({name:"armster"},{$inc :{rat:7}})
MongoServerError: Cannot apply $inc to a value of non-numeric type. {_id: ObjectId('64cb4b45f5bc638d2760131f')} has the field 'rat' of non-numeric type string
testbae> db.details.updateOne({name:"armster"},{$set :{rat:7}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
testbae> db.details.updateOne({name:"armster"},{$inc :{rat:3}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
testbae> db.details.find()
[
  {
    _id: ObjectId("64cb4b45f5bc638d2760131f"),
    name: 'armster',
    add: 'pluto',
    rat: 10
  }
]
testbae> db.details.insertMany({name:"noodles",add:"venus",rat:9},{name:"ice cream",add:"jupiter",rat:9})
MongoInvalidArgumentError: Argument "docs" must be an array of documents
testbae> db.details.insertMany([{name:"noodles",add:"venus",rat:9},{name:"ice cream",add:"jupiter",rat:9}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("64cb4e5ef5bc638d27601320"),
    '1': ObjectId("64cb4e5ef5bc638d27601321")
```

```
  }
}
testbae> db.details.find()
[
  {
    _id: ObjectId("64cb4b45f5bc638d2760131f"),
    name: 'armster',
    add: 'pluto',
    rat: 10
  },
  {
    _id: ObjectId("64cb4e5ef5bc638d27601320"),
    name: 'noodles',
    add: 'venus',
    rat: 9
  },
  {
    _id: ObjectId("64cb4e5ef5bc638d27601321"),
    name: 'ice cream',
    add: 'jupiter',
    rat: 9
  }
]
testbae> db.details.find({rat :{$gt:8}})
[
  {
    _id: ObjectId("64cb4b45f5bc638d2760131f"),
    name: 'armster',
    add: 'pluto',
    rat: 10
  },
  {
    _id: ObjectId("64cb4e5ef5bc638d27601320"),
    name: 'noodles',
    add: 'venus',
    rat: 9
  },
  {
    _id: ObjectId("64cb4e5ef5bc638d27601321"),
    name: 'ice cream',
    add: 'jupiter',
    rat: 9
  }
]
testbae> db.details.find({rat :{$gt:9}})
[
  {
    _id: ObjectId("64cb4b45f5bc638d2760131f"),
```

```
    name: 'armster',
    add: 'pluto',
    rat: 10
  }
]
testbae> db.details.find({rat :{$gt:8,$lt:10}},{_id=0,name=1})
... db.details.find({rat :{$gt:8,$lt:10}},{_id=0,name=1});
Uncaught:
SyntaxError: Invalid shorthand property initializer. (1:42)

> 1 | db.details.find({rat :{$gt:8,$lt:10}},{_id=0,name=1})
    |                                          ^
  2 | db.details.find({rat :{$gt:8,$lt:10}},{_id=0,name=1});
  3 |

testbae> db.details.find({rat :{$gt:8,$lt:10}},{_id:0,name:1})
[ { name: 'noodles' }, { name: 'ice cream' } ]
testbae> db.details.find({name:["ice cream","armster"]})

testbae> db.details.find({name:{$in ["ice cream","armster"]}})
Uncaught:
SyntaxError: Unexpected token, expected "," (1:27)

> 1 | db.details.find({name:{$in ["ice cream","armster"]}})
    |                           ^
  2 |

testbae> db.details.find({name:{$in: ["ice cream","armster"]}})
[
  {
    _id: ObjectId("64cb4b45f5bc638d2760131f"),
    name: 'armster',
    add: 'pluto',
    rat: 10
  },
  {
    _id: ObjectId("64cb4e5ef5bc638d27601321"),
    name: 'ice cream',
    add: 'jupiter',
    rat: 9
  }
]
testbae> db.details.aggregate({$average:{age}}
```

2.Program:

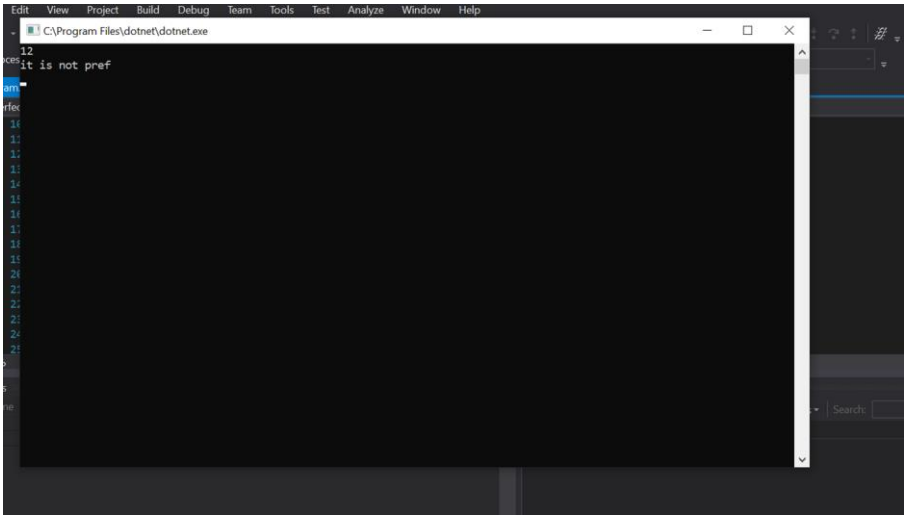
```
using System;

namespace Perfect
{
    class Program
    {
        static void Main(string[] args)
        {
            int num,sum=0;
            num = Convert.ToInt32(Console.ReadLine());
            for (int i=1;i<=num/2;i++)
            {
                if((num%i)==0)
                {
                    //Console.WriteLine(i);
                    sum += i;
                }

            }
            if(num==sum)
            {
                Console.WriteLine("it is pref");
            }
            else
            {
                Console.WriteLine("it is not pref");
            }


            Console.ReadLine();
        }
    }
}
```

output:



3.Program:

```
using System;

namespace prime
{
    class Program
    {
        static void Main(string[] args)
        {
            int num,str,temp =0;

            str = Convert.ToInt32(Console.ReadLine());
            num = Convert.ToInt32(Console.ReadLine());
            for (int i =str; i<= num; i++)
            {
                for(int j =2; j <= i/2; j++)
                {
                    if((i%j) == 0)
                    {
                        temp=0;
                    }
                    else
                    {
                        temp = 1;
                    }
                }
                if(temp == 1)
                {
                    Console.WriteLine(i);
                }
            }
            Console.ReadLine();
        }
    }
}
```

Output: