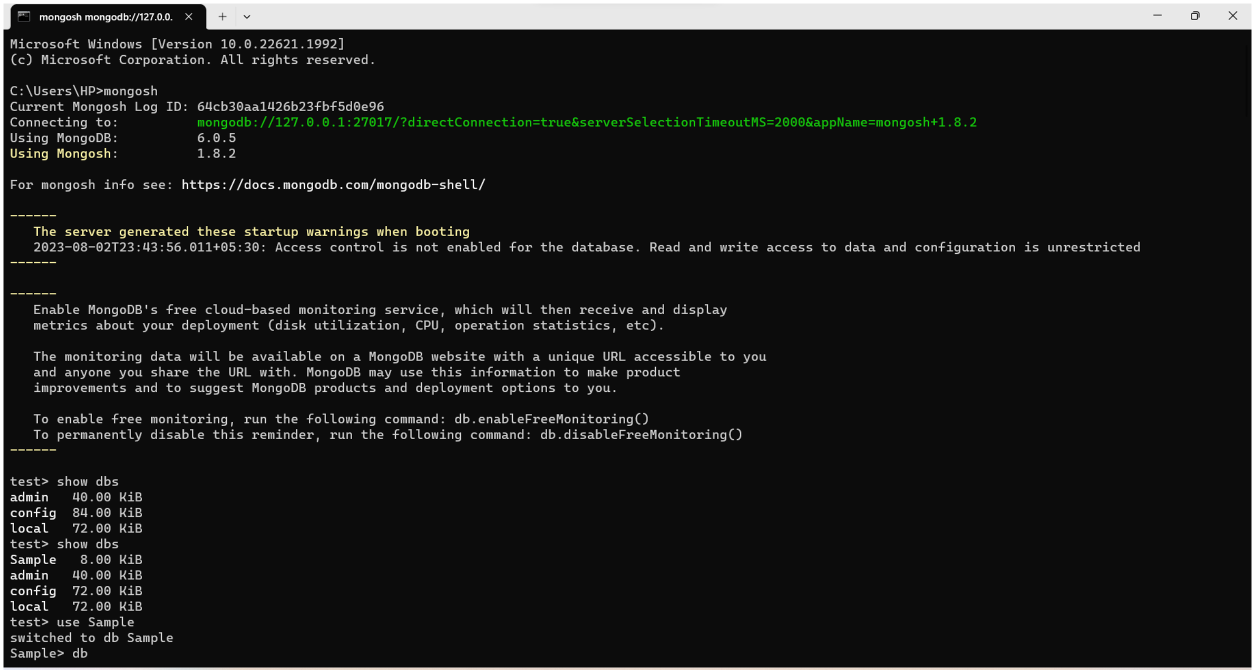
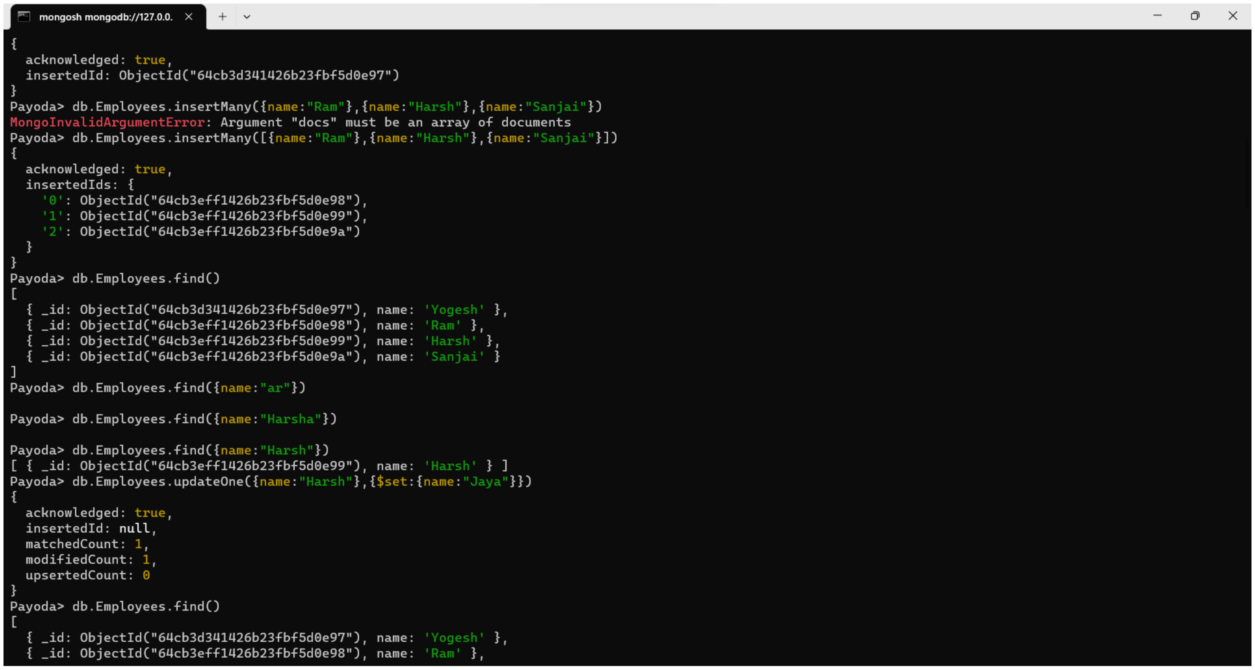
DAY 4:DAILY TASK

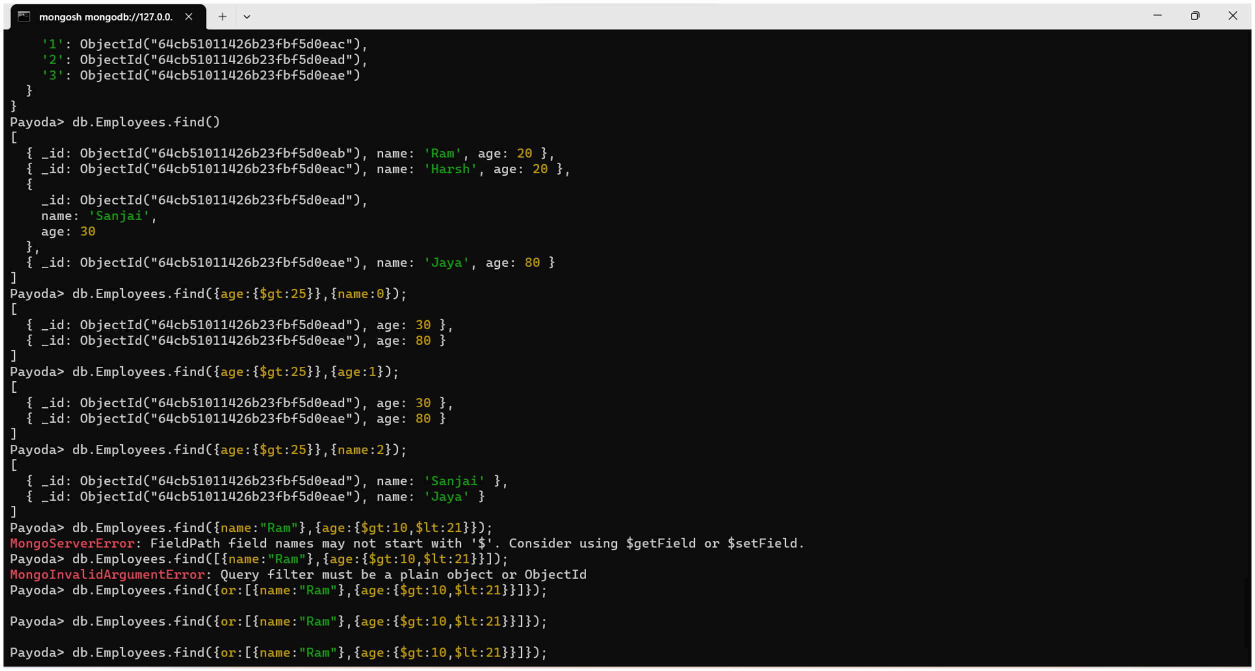
MonogoDB:



```
Microsoft Windows [Version 10.0.22621.1992]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>mongosh
Current Mongosh Log ID: 64cb30aa1426b23fbf5d0e96
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1.8.2
Using MongoDB:          6.0.5
Using Mongosh:          1.8.2

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

------
   The server generated these startup warnings when booting
   2023-08-02T23:43:56.011+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
------

------
   Enable MongoDB's free cloud-based monitoring service, which will then receive and display
   metrics about your deployment (disk utilization, CPU, operation statistics, etc).

   The monitoring data will be available on a MongoDB website with a unique URL accessible to you
   and anyone you share the URL with. MongoDB may use this information to make product
   improvements and to suggest MongoDB products and deployment options to you.

   To enable free monitoring, run the following command: db.enableFreeMonitoring()
   To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
------

test> show dbs
admin   40.00 KiB
config  84.00 KiB
local   72.00 KiB
test> show dbs
Sample   8.00 KiB
admin   40.00 KiB
config  72.00 KiB
local   72.00 KiB
test> use Sample
switched to db Sample
Sample> db
```



```
{
  acknowledged: true,
  insertedId: ObjectId("64cb3d341426b23fbf5d0e97")
}
Payoda> db.Employees.insertMany({name:"Ram"},{name:"Harsh"},{name:"Sanjai"})
MongoInvalidArgumentError: Argument "docs" must be an array of documents
Payoda> db.Employees.insertMany([{name:"Ram"},{name:"Harsh"},{name:"Sanjai"}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("64cb3eff1426b23fbf5d0e98"),
    '1': ObjectId("64cb3eff1426b23fbf5d0e99"),
    '2': ObjectId("64cb3eff1426b23fbf5d0e9a")
  }
}
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb3d341426b23fbf5d0e97"), name: 'Yogesh' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e98"), name: 'Ram' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e99"), name: 'Harsh' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e9a"), name: 'Sanjai' }
]
Payoda> db.Employees.find({name:"ar"})

Payoda> db.Employees.find({name:"Harsha"})

Payoda> db.Employees.find({name:"Harsh"})
[ { _id: ObjectId("64cb3eff1426b23fbf5d0e99"), name: 'Harsh' } ]
Payoda> db.Employees.updateOne({name:"Harsh"},{$set:{name:"Jaya"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb3d341426b23fbf5d0e97"), name: 'Yogesh' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e98"), name: 'Ram' },
```



```
    '1': ObjectId("64cb51011426b23fbf5d0eac"),
    '2': ObjectId("64cb51011426b23fbf5d0ead"),
    '3': ObjectId("64cb51011426b23fbf5d0eae")
  }
}
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb51011426b23fbf5d0eab"), name: 'Ram', age: 20 },
  { _id: ObjectId("64cb51011426b23fbf5d0eac"), name: 'Harsh', age: 20 },
  {
    _id: ObjectId("64cb51011426b23fbf5d0ead"),
    name: 'Sanjai',
    age: 30
  },
  { _id: ObjectId("64cb51011426b23fbf5d0eae"), name: 'Jaya', age: 80 }
]
Payoda> db.Employees.find({age:{$gt:25}},{name:0});
[
  { _id: ObjectId("64cb51011426b23fbf5d0ead"), age: 30 },
  { _id: ObjectId("64cb51011426b23fbf5d0eae"), age: 80 }
]
Payoda> db.Employees.find({age:{$gt:25}},{age:1});
[
  { _id: ObjectId("64cb51011426b23fbf5d0ead"), age: 30 },
  { _id: ObjectId("64cb51011426b23fbf5d0eae"), age: 80 }
]
Payoda> db.Employees.find({age:{$gt:25}},{name:2});
[
  { _id: ObjectId("64cb51011426b23fbf5d0ead"), name: 'Sanjai' },
  { _id: ObjectId("64cb51011426b23fbf5d0eae"), name: 'Jaya' }
]
Payoda> db.Employees.find({name:"Ram"},{age:{$gt:10,$lt:21}});
MongoServerError: FieldPath field names may not start with '$'. Consider using $getField or $setField.
Payoda> db.Employees.find([{name:"Ram"},{age:{$gt:10,$lt:21}}]);
MongoInvalidArgumentError: Query filter must be a plain object or ObjectId
Payoda> db.Employees.find({or:[{name:"Ram"},{age:{$gt:10,$lt:21}}]});

Payoda> db.Employees.find({or:[{name:"Ram"},{age:{$gt:10,$lt:21}}]});

Payoda> db.Employees.find({or:[{name:"Ram"},{age:{$gt:10,$lt:21}}]});
```

Shell commands:

```
Microsoft Windows [Version 10.0.22621.1992]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>mongosh
Current Mongosh Log ID: 64cb30aa1426b23fbf5d0e96
Connecting to:
mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1.8.2
Using MongoDB:          6.0.5
Using Mongosh:          1.8.2

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

------
   The server generated these startup warnings when booting
   2023-08-02T23:43:56.011+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
------

------
   Enable MongoDB's free cloud-based monitoring service, which will then receive and display
   metrics about your deployment (disk utilization, CPU, operation statistics, etc).

   The monitoring data will be available on a MongoDB website with a unique URL accessible to
you
   and anyone you share the URL with. MongoDB may use this information to make product
   improvements and to suggest MongoDB products and deployment options to you.

   To enable free monitoring, run the following command: db.enableFreeMonitoring()
   To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
------

test> show dbs
admin   40.00 KiB
config  84.00 KiB
local   72.00 KiB
test> show dbs
Sample   8.00 KiB
admin   40.00 KiB
config  72.00 KiB
local   72.00 KiB
test> use Sample
switched to db Sample
Sample> db
Sample
Sample> use Payoda
switched to db Payoda
Payoda> bd.createCollection("Employees");
ReferenceError: bd is not defined
Payoda> db.createCollection("Employees");
{ ok: 1 }
Payoda> db.Employees.insertOne({name:"Yogesh"})
{
  acknowledged: true,
  insertedId: ObjectId("64cb3d341426b23fbf5d0e97")
}
Payoda> db.Employees.insertMany({name:"Ram"},{name:"Harsh"},{name:"Sanjai"})
MongoInvalidArgumentError: Argument "docs" must be an array of documents
Payoda> db.Employees.insertMany([{name:"Ram"},{name:"Harsh"},{name:"Sanjai"}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("64cb3eff1426b23fbf5d0e98"),
    '1': ObjectId("64cb3eff1426b23fbf5d0e99"),
    '2': ObjectId("64cb3eff1426b23fbf5d0e9a")
  }
}
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb3d341426b23fbf5d0e97"), name: 'Yogesh' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e98"), name: 'Ram' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e99"), name: 'Harsh' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e9a"), name: 'Sanjai' }
]
Payoda> db.Employees.find({name:"ar"})

Payoda> db.Employees.find({name:"Harsha"})

Payoda> db.Employees.find({name:"Harsh"})
[ { _id: ObjectId("64cb3eff1426b23fbf5d0e99"), name: 'Harsh' } ]
Payoda> db.Employees.updateOne({name:"Harsh"},{$set:{name:"Jaya"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb3d341426b23fbf5d0e97"), name: 'Yogesh' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e98"), name: 'Ram' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e99"), name: 'Jaya' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e9a"), name: 'Sanjai' }
]
Payoda> db.Employees.updateOne({name:"Jaya"},{$set:{name:"Jayan"}},{upsert:true})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb3d341426b23fbf5d0e97"), name: 'Yogesh' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e98"), name: 'Ram' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e99"), name: 'Jayan' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e9a"), name: 'Sanjai' }
```

```
]
Payoda> db.Employees.updateOne({name:"Jaya"},{$set:{name:"Jayan"}},{upsert:true})
{
  acknowledged: true,
  insertedId: ObjectId("64cb4a116d206c62ba7f4e3d"),
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 1
}
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb3d341426b23fbf5d0e97"), name: 'Yogesh' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e98"), name: 'Ram' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e99"), name: 'Jayan' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e9a"), name: 'Sanjai' },
  { _id: ObjectId("64cb4a116d206c62ba7f4e3d"), name: 'Jayan' }
]
Payoda> db.Employees.deleteOne({name:"Jayan"})
{ acknowledged: true, deletedCount: 1 }
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb3d341426b23fbf5d0e97"), name: 'Yogesh' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e98"), name: 'Ram' },
  { _id: ObjectId("64cb3eff1426b23fbf5d0e9a"), name: 'Sanjai' },
  { _id: ObjectId("64cb4a116d206c62ba7f4e3d"), name: 'Jayan' }
]
Payoda> db.Employees.deleteMany()
MongoshInvalidInputError: [COMMON-10001] Missing required argument at position 0
(Collection.deleteMany)
Payoda> db.Employees.deleteMany({})
{ acknowledged: true, deletedCount: 4 }
Payoda> db.Employees.find()

Payoda>
db.Employees.insertMany([{name:"Ram"},{name:"Harsh"},{name:"Sanjai"},{name:"Jaya"}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("64cb4ae01426b23fbf5d0e9b"),
    '1': ObjectId("64cb4ae01426b23fbf5d0e9c"),
    '2': ObjectId("64cb4ae01426b23fbf5d0e9d"),
    '3': ObjectId("64cb4ae01426b23fbf5d0e9e")
  }
}
```

```
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9b"), name: 'Ram' },
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9c"), name: 'Harsh' },
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9d"), name: 'Sanjai' },
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9e"), name: 'Jaya' }
]
Payoda>
db.Employees.insertMany([{name:"Ram",age:20},{name:"Harsh",age:20},{name:"Sanjai",age:30
},{name:"Jaya",age:80}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("64cb4be61426b23fbf5d0e9f"),
    '1': ObjectId("64cb4be61426b23fbf5d0ea0"),
    '2': ObjectId("64cb4be61426b23fbf5d0ea1"),
    '3': ObjectId("64cb4be61426b23fbf5d0ea2")
  }
}
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9b"), name: 'Ram' },
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9c"), name: 'Harsh' },
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9d"), name: 'Sanjai' },
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9e"), name: 'Jaya' },
  { _id: ObjectId("64cb4be61426b23fbf5d0e9f"), name: 'Ram', age: 20 },
  { _id: ObjectId("64cb4be61426b23fbf5d0ea0"), name: 'Harsh', age: 20 },
  {
    _id: ObjectId("64cb4be61426b23fbf5d0ea1"),
    name: 'Sanjai',
    age: 30
  },
  { _id: ObjectId("64cb4be61426b23fbf5d0ea2"), name: 'Jaya', age: 80 }
]
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9b"), name: 'Ram' },
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9c"), name: 'Harsh' },
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9d"), name: 'Sanjai' },
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9e"), name: 'Jaya' },
  { _id: ObjectId("64cb4be61426b23fbf5d0e9f"), name: 'Ram', age: 20 },
  { _id: ObjectId("64cb4be61426b23fbf5d0ea0"), name: 'Harsh', age: 20 },
  {
    _id: ObjectId("64cb4be61426b23fbf5d0ea1"),
    name: 'Sanjai',
    age: 30
  },
  { _id: ObjectId("64cb4be61426b23fbf5d0ea2"), name: 'Jaya', age: 80 }
]
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9b"), name: 'Ram' },
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9c"), name: 'Harsh' },
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9d"), name: 'Sanjai' },
  { _id: ObjectId("64cb4ae01426b23fbf5d0e9e"), name: 'Jaya' },
  { _id: ObjectId("64cb4be61426b23fbf5d0e9f"), name: 'Ram', age: 20 },
  { _id: ObjectId("64cb4be61426b23fbf5d0ea0"), name: 'Harsh', age: 20 },
  {
    _id: ObjectId("64cb4be61426b23fbf5d0ea1"),
    name: 'Sanjai',
    age: 30
  },
  { _id: ObjectId("64cb4be61426b23fbf5d0ea2"), name: 'Jaya', age: 80 }
]
Payoda> db.Employees.deleteMany({})
{ acknowledged: true, deletedCount: 8 }
Payoda>
db.Employees.insertMany([{name:"Ram",age:20},{name:"Harsh",age:20},{name:"Sanjai",age:30
},{name:"Jaya",age:80}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("64cb4c0a1426b23fbf5d0ea3"),
    '1': ObjectId("64cb4c0a1426b23fbf5d0ea4"),
    '2': ObjectId("64cb4c0a1426b23fbf5d0ea5"),
    '3': ObjectId("64cb4c0a1426b23fbf5d0ea6")
  }
}
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb4c0a1426b23fbf5d0ea3"), name: 'Ram', age: 20 },
  { _id: ObjectId("64cb4c0a1426b23fbf5d0ea4"), name: 'Harsh', age: 20 },
  {
    _id: ObjectId("64cb4c0a1426b23fbf5d0ea5"),
    name: 'Sanjai',
    age: 30
  },
  { _id: ObjectId("64cb4c0a1426b23fbf5d0ea6"), name: 'Jaya', age: 80 }
```

```
]
Payoda> db.Employees.find({age:{$gt:25}});

Payoda> show dbs
Payoda  72.00 KiB
Sample  40.00 KiB
admin   40.00 KiB
config  96.00 KiB
local   72.00 KiB
Payoda> use Payoda
already on db Payoda
Payoda>
db.Employees.insertMany([{name:"Ram",age:20},{name:"Harsh",age:20},{name:"Sanjai",age:30
},{name:"Jaya",age:80}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("64cb50ea1426b23fbf5d0ea7"),
    '1': ObjectId("64cb50ea1426b23fbf5d0ea8"),
    '2': ObjectId("64cb50ea1426b23fbf5d0ea9"),
    '3': ObjectId("64cb50ea1426b23fbf5d0eaa")
  }
}
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb4c0a1426b23fbf5d0ea3"), name: 'Ram', age: 20 },
  { _id: ObjectId("64cb4c0a1426b23fbf5d0ea4"), name: 'Harsh', age: 20 },
  {
    _id: ObjectId("64cb4c0a1426b23fbf5d0ea5"),
    name: 'Sanjai',
    age: 30
  },
  { _id: ObjectId("64cb4c0a1426b23fbf5d0ea6"), name: 'Jaya', age: 80 },
  { _id: ObjectId("64cb50ea1426b23fbf5d0ea7"), name: 'Ram', age: 20 },
  { _id: ObjectId("64cb50ea1426b23fbf5d0ea8"), name: 'Harsh', age: 20 },
  {
    _id: ObjectId("64cb50ea1426b23fbf5d0ea9"),
    name: 'Sanjai',
    age: 30
  },
  { _id: ObjectId("64cb50ea1426b23fbf5d0eaa"), name: 'Jaya', age: 80 }
]
Payoda> db.Employees.deleteMany({})
{ acknowledged: true, deletedCount: 8 }
```

```
Payoda>
db.Employees.insertMany([{name:"Ram",age:20},{name:"Harsh",age:20},{name:"Sanjai",age:30
},{name:"Jaya",age:80}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("64cb51011426b23fbf5d0eab"),
    '1': ObjectId("64cb51011426b23fbf5d0eac"),
    '2': ObjectId("64cb51011426b23fbf5d0ead"),
    '3': ObjectId("64cb51011426b23fbf5d0eae")
  }
}
Payoda> db.Employees.find()
[
  { _id: ObjectId("64cb51011426b23fbf5d0eab"), name: 'Ram', age: 20 },
  { _id: ObjectId("64cb51011426b23fbf5d0eac"), name: 'Harsh', age: 20 },
  {
    _id: ObjectId("64cb51011426b23fbf5d0ead"),
    name: 'Sanjai',
    age: 30
  },
  { _id: ObjectId("64cb51011426b23fbf5d0eae"), name: 'Jaya', age: 80 }
]
Payoda> db.Employees.find({age:{$gt:25}},{name:0});
[
  { _id: ObjectId("64cb51011426b23fbf5d0ead"), age: 30 },
  { _id: ObjectId("64cb51011426b23fbf5d0eae"), age: 80 }
]
Payoda> db.Employees.find({age:{$gt:25}},{age:1});
[
  { _id: ObjectId("64cb51011426b23fbf5d0ead"), age: 30 },
  { _id: ObjectId("64cb51011426b23fbf5d0eae"), age: 80 }
]
Payoda> db.Employees.find({age:{$gt:25}},{name:2});
[
  { _id: ObjectId("64cb51011426b23fbf5d0ead"), name: 'Sanjai' },
  { _id: ObjectId("64cb51011426b23fbf5d0eae"), name: 'Jaya' }
]
Payoda> db.Employees.find({name:"Ram"},{age:{$gt:10,$lt:21}});
MongoServerError: FieldPath field names may not start with '$'. Consider using $getField or
$setField.
Payoda> db.Employees.find([{name:"Ram"},{age:{$gt:10,$lt:21}}]);
MongoInvalidArgumentError: Query filter must be a plain object or ObjectId
Payoda> db.Employees.find({or:[{name:"Ram"},{age:{$gt:10,$lt:21}}]});
Payoda> db.Employees.find({or:[{name:"Ram"},{age:{$gt:10,$lt:21}}]});
```

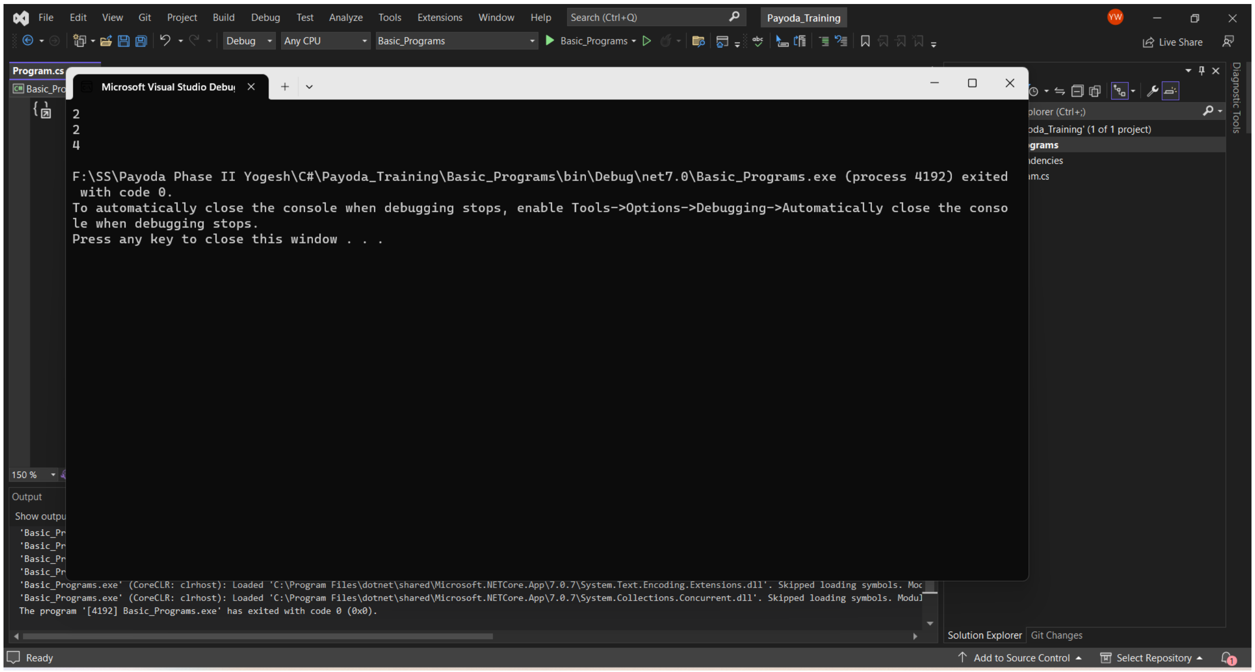
```
Payoda> db.Employees.find({or:[{name:"Ram"},{age:{$gt:10,$lt:21}}]});
```

C# programs

Program 1:

```
int num1, num2, rem;

num1 = Convert.ToInt32(Console.ReadLine());
num2 = Convert.ToInt32(Console.ReadLine());

rem = num1 + num2;

Console.WriteLine(rem);
```

Output:



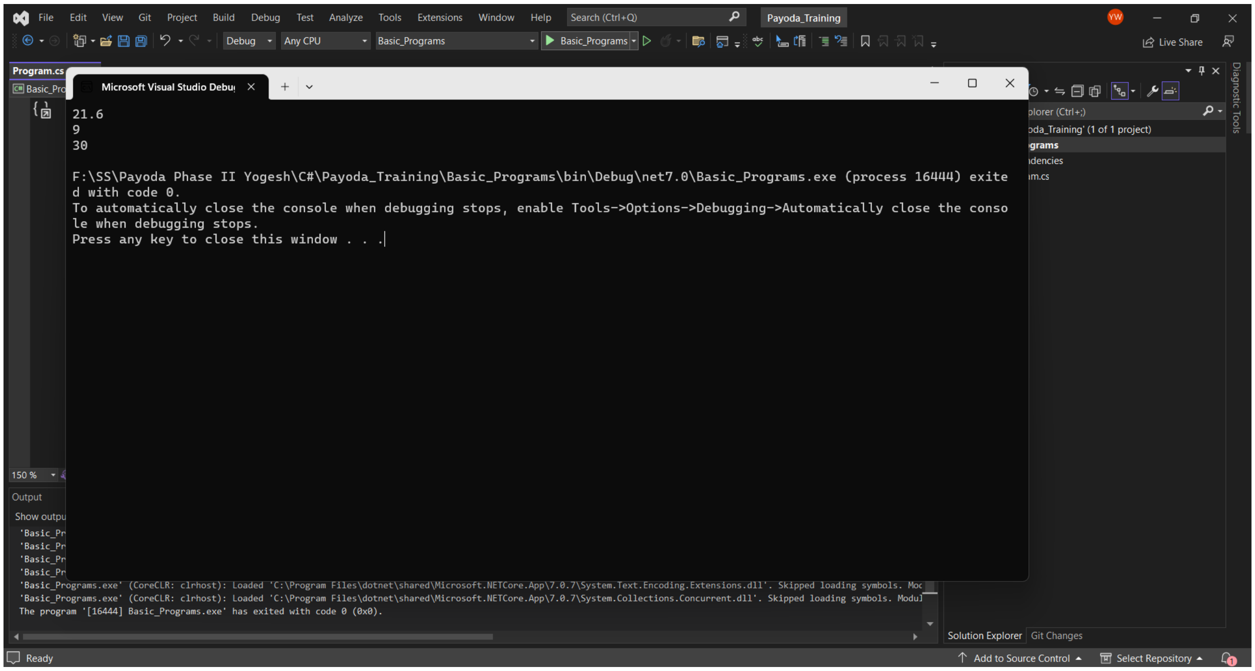
```
2
2
4

F:\SS\Payoda Phase II Yogesh\C#\Payoda_Training\Basic_Programs\bin\Debug\net7.0\Basic_Programs.exe (process 4192) exited
 with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

```
'Basic_Pr
'Basic_Pr
'Basic_Pr
'Basic_Pr
'Basic_Programs.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\7.0.7\System.Text.Encoding.Extensions.dll'. Skipped loading symbols. Moc
'Basic_Programs.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\7.0.7\System.Collections.Concurrent.dll'. Skipped loading symbols. Modul
The program '[4192] Basic_Programs.exe' has exited with code 0 (0x0).
```

Program 2:

```
double num1;

int num2, rem;

num1 = Convert.ToDouble(Console.ReadLine());
num2 = Convert.ToInt32(Console.ReadLine());

rem = (int)num1 + num2;

Console.WriteLine(rem);
```

Output:

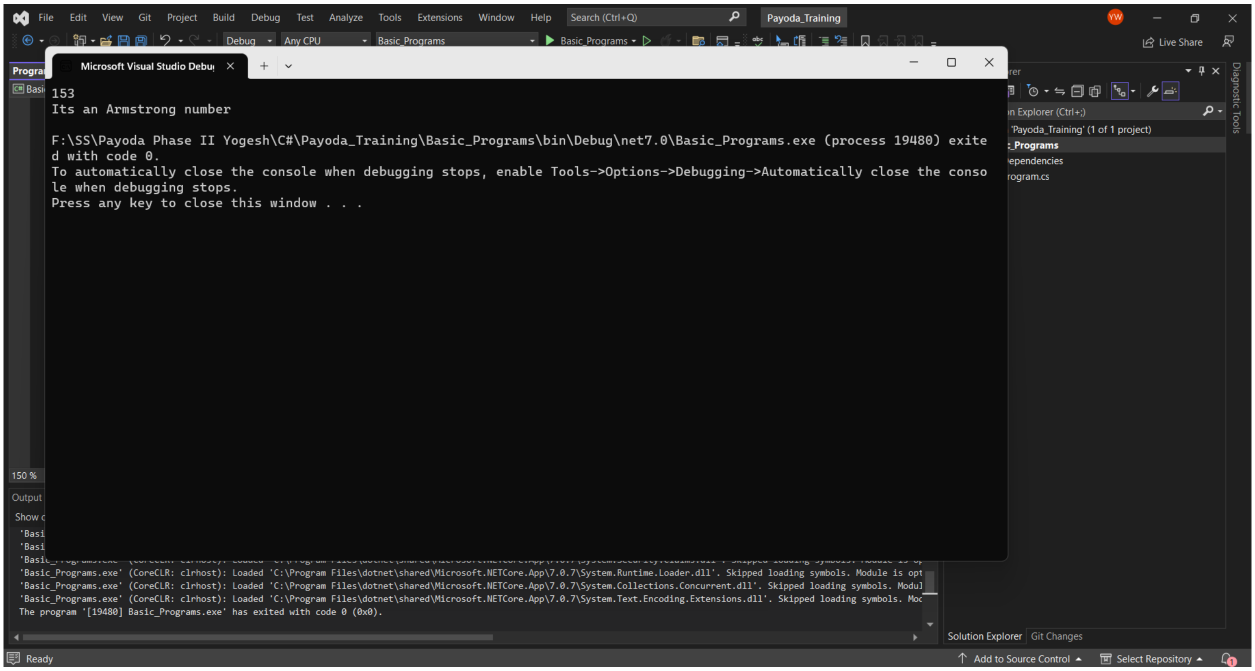


```
21.6
9
30

F:\SS\Payoda Phase II Yogesh\C#\Payoda_Training\Basic_Programs\bin\Debug\net7.0\Basic_Programs.exe (process 16444) exite
d with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

```
'Basic_Pr
'Basic_Pr
'Basic_Pr
'Basic_Pr
'Basic_Programs.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\7.0.7\System.Text.Encoding.Extensions.dll'. Skipped loading symbols. Moc
'Basic_Programs.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\7.0.7\System.Collections.Concurrent.dll'. Skipped loading symbols. Modul
The program '[16444] Basic_Programs.exe' has exited with code 0 (0x0).
```

Program 3:

```
//Armstrong Number


int num, sum = 0;

num=Convert.ToInt32(Console.ReadLine());
int temp = num;

while(num > 0)
{
    int rem = num % 10;
    sum += rem * rem * rem;
    num /= 10;
}

if (sum == temp)
    Console.WriteLine("Its an Armstrong number");
else Console.WriteLine("Not an Armstrong number");
```



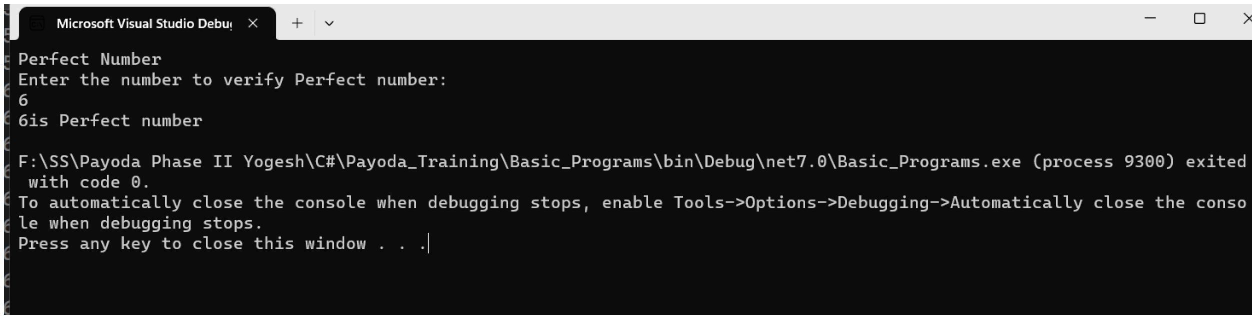
```
153
Its an Armstrong number

F:\SS\Payoda Phase II Yogesh\C#\Payoda_Training\Basic_Programs\bin\Debug\net7.0\Basic_Programs.exe (process 19480) exite
d with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

```
'Basic_Programs.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\7.0.7\System.Runtime.Loader.dll'. Skipped loading symbols. Module is opt
'Basic_Programs.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\7.0.7\System.Collections.Concurrent.dll'. Skipped loading symbols. Modul
'Basic_Programs.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\7.0.7\System.Text.Encoding.Extensions.dll'. Skipped loading symbols. Moc
The program '[19480] Basic_Programs.exe' has exited with code 0 (0x0).
```

Program 4:

To find Perfect number

```
//To find perfect number

int num1, sum = 0;
Console.WriteLine("Perfect Number");
Console.WriteLine("Enter the number to verify Perfect number: ");
num1 = Convert.ToInt32(Console.ReadLine());

int i = 1;
while(i<=num1/2)
{
   if (num1 % i == 0)
         {
      sum = sum + i;
   }
   i++;
}
if (sum == num1)
{
   Console.WriteLine( num1+"is Perfect number");
}
else
{
   Console.WriteLine(num1 + "is not a Perfect Number");
}
```

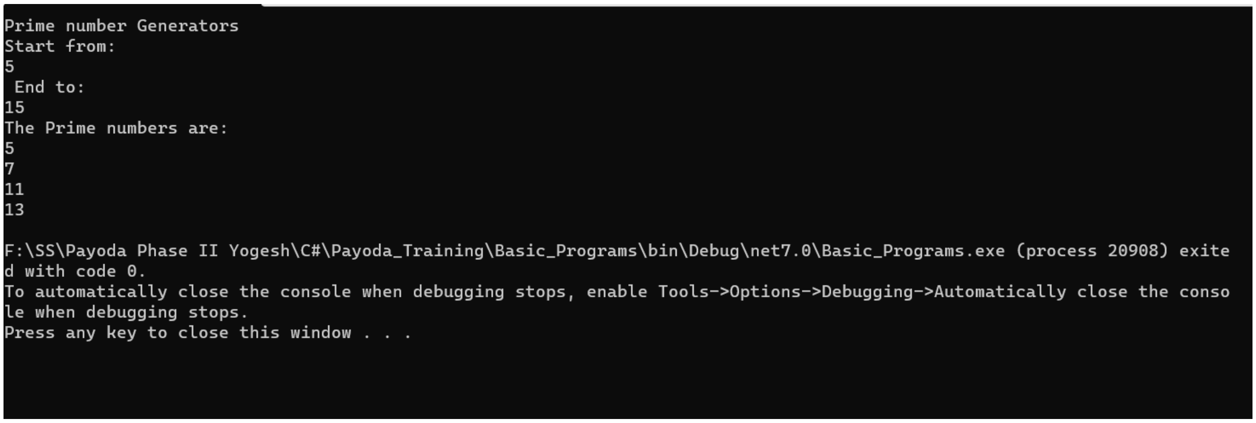


```
Perfect Number
Enter the number to verify Perfect number:
6
6is Perfect number

F:\SS\Payoda Phase II Yogesh\C#\Payoda_Training\Basic_Programs\bin\Debug\net7.0\Basic_Programs.exe (process 9300) exited
 with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

Program 6:

```
// Generate Prime number between values.

int num1, num2;
Console.WriteLine("Prime number Generators");
Console.WriteLine("Start from: ");
num1 = Convert.ToInt32(Console.ReadLine());
Console.WriteLine(" End to: ");
num2 = Convert.ToInt32(Console.ReadLine());
Console.WriteLine("The Prime numbers are: ");
for (int i = num1; i <= num2; i++)
{
    int val = 0;
    int j = 2;
    while(j<=i/2)
    {
        if (i % j == 0)
        {
            val = 1;
        }
        j++;
    }
    if (val == 0)
    {
        Console.WriteLine(i);
    }
}
```

Output:



```
Prime number Generators
Start from:
5
 End to:
15
The Prime numbers are:
5
7
11
13

F:\SS\Payoda Phase II Yogesh\C#\Payoda_Training\Basic_Programs\bin\Debug\net7.0\Basic_Programs.exe (process 20908) exite
d with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```