coviddef1.h

```cpp
#include <iostream>
#include <string>
using namespace std;

unsigned int mystoi(string str)
{
  short digs[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
  unsigned int intval = 0;
  int pos = 1, index;
  for (int i = str.length() - 1; i >= 0; i--)
  {
    switch (str[i])
    {
    case '0':
      index = 0;
      break;
    case '1':
      index = 1;
      break;
    case '2':
      index = 2;
      break;
    case '3':
      index = 3;
      break;
    case '4':
      index = 4;
      break;
    case '5':
      index = 5;
      break;
    case '6':
      index = 6;
      break;
    case '7':
      index = 7;
      break;
    case '8':
      index = 8;
      break;
    case '9':
      index = 9;
      break;
    default:
    {
      cerr << " Not a digit error \n";
      return -2;
    }
    }; // end of switch
    intval = intval + digs[index] * pos;
    // cerr << " intval = " << intval << " pos = " << pos
    //      << " digs[" << index << " ] = " << digs[index] << endl;
```

```
    pos = 10 * pos;
  }
  return intval;
}

class coviddata
{ // data members are all public here
public:
  string state;
  unsigned int population;
  unsigned int confirmed;
  unsigned int active;
  unsigned int recovered;
  unsigned int deceased;
  unsigned int tested;
  unsigned int vaccinated;
  float vacc2popratio; // 3 ratio asked in Lab 2 problem 3
  float rec2cnfratio;
  float tst2popratio;
  // ..... for future extensions
  // function members are also public
  coviddata()
  {
    state = "";
    population = confirmed = active = recovered = 0;
    deceased = tested = vaccinated = 0;
    vacc2popratio = rec2cnfratio = tst2popratio = 0.0;
  }
  friend ostream &operator<<(ostream &, coviddata);
  friend istream &operator>>(istream &, coviddata &);
};

// non-member friend functions for i/o on coviddata objects

ostream &operator<<(ostream &x, coviddata cdata)
{
  x << " All 10 attributes of Covid Data for state " << cdata.state << endl;
  //   for (int i = 0; i < 10; i++)
  x << " Population : " << cdata.population << " "
    << " Confirmed Cases : " << cdata.confirmed << endl
    << " Active Cases : " << cdata.active << " "
    << " Recovered Cases : " << cdata.recovered << endl;
  x << " Deceased Cases : " << cdata.deceased << " "
    << " Tested Cases  : " << cdata.tested << " "
    << " Vaccinated Cases : " << cdata.vaccinated << endl;
  x << " Vaccinated to Population Ratio : " << cdata.vacc2popratio << endl
    << " Recovered to Confirmed Ratio : " << cdata.rec2cnfratio << endl
    << " Tested to Population Ratio : " << cdata.tst2popratio << endl;
  return x;
}
```

```
istream &operator>>(istream &x, covidData &cdata)
{
  string statedata;
  getline(cin, statedata); // read the entire line in a string
  short k = 0, count = 0;
  string first;
  while ((k = statedata.find(',')) != 0 && k <= statedata.length())
  { // comma found at position k
    first = statedata.substr(0, k);
    first = statedata.substr(0, k);
    statedata = statedata.substr(k + 1, statedata.length() - k - 1);
    switch (count)
    {
    case 0:
    {
      cdata.state = first;
      break;
    }
    case 1:
    {
      cdata.population = mystoi(first) * 100000;
      break;
    }
    case 2:
    {
      cdata.confirmed = mystoi(first);
      break;
    }
    case 3:
    {
      cdata.active = mystoi(first);
      break;
    }
    case 4:
    {
      cdata.recovered = mystoi(first);
      break;
    }
    case 5:
    {
      cdata.deceased = mystoi(first);
      break;
    }
    case 6:
    {
      cdata.tested = 100000 * mystoi(first);
      break;
    }
    case 7:
    {
    } // return -2;}
    }; // end of inner while : extracted one field at each iteration
```

```
    count++;
  };
  if (statedata.length() > 0)
  {
    cdata.vaccinated = 100000 * mystoi(statedata); // last data value
  };
  return x;
}
```

coviduse1.C

```
#include <iostream>
#include <string>
#include "coviddef1.h"
int main()
{
  string statedata;
  coviddata cvdata;
  cin >> cvdata;          // read one line
  cout << cvdata << endl; // display object
  return 0;
}
```

coviddef2.h

```cpp
#include <iostream>
#include <string>
    using namespace std;

unsigned int mystoi(string str)
{
  short digs[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
  unsigned int intval = 0;
  int pos = 1, index;
  for (int i = str.length() - 1; i >= 0; i--)
  {
    switch (str[i])
    {
    case '0':
      index = 0;
      break;
    case '1':
      index = 1;
      break;
    case '2':
      index = 2;
      break;
    case '3':
      index = 3;
      break;
    case '4':
      index = 4;
      break;
    case '5':
      index = 5;
      break;
    case '6':
      index = 6;
      break;
    case '7':
      index = 7;
      break;
    case '8':
      index = 8;
      break;
    case '9':
      index = 9;
      break;
    default:
    {
      cerr << " Not a digit error \n";
      return -2;
    }
    }; // end of switch
    intval = intval + digs[index] * pos;
    // cerr << " intval = " << intval << " pos = " << pos
    //      << " digs[" << index << " ] = " << digs[index] << endl;
```

```
    pos = 10 * pos;
  }
  return intval;
}

class coviddata
{ // data members are all public here
public:
  string state;
  unsigned int population;
  unsigned int confirmed;
  unsigned int active;
  unsigned int recovered;
  unsigned int deceased;
  unsigned int tested;
  unsigned int vaccinated;
  float vacc2popratio; // 3 ratio asked in Lab 2 problem 3
  float rec2cnfratio;
  float tst2popratio;
  // ..... for future extensions
  // function members are also public

  void ratios()
  {
    vacc2popratio = (float)vaccinated / population;
    rec2cnfratio = (float)recovered / confirmed;
    tst2popratio = (float)tested / population;
  }

  coviddata()
  {
    state = "";
    population = confirmed = active = recovered = 0;
    deceased = tested = vaccinated = 0;
    vacc2popratio = rec2cnfratio = tst2popratio = 0.0;
  }
  friend ostream &operator<<(ostream &, coviddata);
  friend istream &operator>>(istream &, coviddata &);
};

// non-member friend functions for i/o on coviddata objects

ostream &operator<<(ostream &x, coviddata cdata)
{
  x << " All 10 attributes of Covid Data for state " << cdata.state << endl;
  //   for (int i = 0; i < 10; i++)
  x << " Population : " << cdata.population << " "
    << " Confirmed Cases : " << cdata.confirmed << endl
    << " Active Cases : " << cdata.active << " "
    << " Recovered Cases : " << cdata.recovered << endl;
  x << " Deceased Cases : " << cdata.deceased << " "
```

```
    <<   Tested Cases  :   << cdata.tested <<
    << " Vaccinated Cases : " << cdata.vaccinated << endl;
  x << " Vaccinated to Population Ratio : " << cdata.vacc2popratio << endl
    << " Recovered to Confirmed Ratio : " << cdata.rec2cnfratio << endl
    << " Tested to Population Ratio : " << cdata.tst2popratio << endl;
  return x;
}
istream &operator>>(istream &x, coviddata &cdata)
{
  string statedata;
  getline(cin, statedata); // read the entire line in a string
  short k = 0, count = 0;
  string first;
  while ((k = statedata.find(',')) != 0 && k <= statedata.length())
  { // comma found at position k
    first = statedata.substr(0, k);
    statedata = statedata.substr(k + 1, statedata.length() - k - 1);
    switch (count)
    {
    case 0:
    {
      cdata.state = first;
      break;
    }
    case 1:
    {
      cdata.population = mystoi(first) * 100000;
      break;
    }
    case 2:
    {
      cdata.confirmed = mystoi(first);
      break;
    }
    case 3:
    {
      cdata.active = mystoi(first);
      break;
    }
    case 4:
    {
      cdata.recovered = mystoi(first);
      break;
    }
    case 5:
    {
      cdata.deceased = mystoi(first);
      break;
    }
    case 6:
    {
      cdata.tested = 100000 * mystoi(first);
```

```
        break;
      }
      case 7:
      {
      } // return -2;}
      }; // end of inner while : extracted one field at each iteration
      count++;
    };
    if (statedata.length() > 0)
    {
      cdata.vaccinated = 100000 * mystoi(statedata); // last data value
    };
    return x;
  }
```

coviduse2.cpp

```
  #include <iostream>
  #include <string>
  #include "coviddef2.h"

  int main()
  {
      string statedata;
      coviddata cvdata;
      cin >> cvdata;
      cvdata.ratios();
      cout << cvdata << endl;
      return 0;
  }
```

coviddef3.h

```cpp
#include <iostream>
#include <string>
using namespace std;

unsigned int mystoi(string str)
{
    short digs[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
    unsigned int intval = 0;
    int pos = 1, index;
    for (int i = str.length() - 1; i >= 0; i--)
    {
        switch (str[i])
        {
        case '0':
            index = 0;
            break;
        case '1':
            index = 1;
            break;
        case '2':
            index = 2;
            break;
        case '3':
            index = 3;
            break;
        case '4':
            index = 4;
            break;
        case '5':
            index = 5;
            break;
        case '6':
            index = 6;
            break;
        case '7':
            index = 7;
            break;
        case '8':
            index = 8;
            break;
        case '9':
            index = 9;
            break;
        default:
        {
            cerr << " Not a digit error \n";
            return -2;
        }
        }; // end of switch
        intval = intval + digs[index] * pos;
        // cerr << " intval = " << intval << " pos = " << pos
        //      << " digs[" << index << " ] = " << digs[index] << endl;
```

```
        pos = 10 * pos;
    }
    return intval;
}

class coviddata
{ // data members are all public here
public:
    string state;
    unsigned int population;
    unsigned int confirmed;
    unsigned int active;
    unsigned int recovered;
    unsigned int deceased;
    unsigned int tested;
    unsigned int vaccinated;
    float vacc2popratio; // 3 ratio asked in Lab 2 problem 3
    float rec2cnfratio;
    float tst2popratio;
    void ratios()
    {
        vacc2popratio = (float)vaccinated / population;
        rec2cnfratio = (float)recovered / confirmed;
        tst2popratio = (float)tested / population;
    }

    coviddata()
    {
        state = "";
        population = confirmed = active = recovered = 0;
        deceased = tested = vaccinated = 0;
        vacc2popratio = rec2cnfratio = tst2popratio = 0.0;
    }
    friend ostream &operator<<(ostream &, coviddata);
    friend istream &operator>>(istream &, coviddata &);
};

// non-member friend functions for i/o on coviddata objects

ostream &operator<<(ostream &x, coviddata cdata)
{
    x << " All 10 attributes of Covid Data for state " << cdata.state << endl;
    //   for (int i = 0; i < 10; i++)
    x << " Population : " << cdata.population << " "
      << " Confirmed Cases : " << cdata.confirmed << endl
      << " Active Cases : " << cdata.active << " "
      << " Recovered Cases : " << cdata.recovered << endl;
    x << " Deceased Cases : " << cdata.deceased << " "
      << " Tested Cases  : " << cdata.tested << " "
      << " Vaccinated Cases : " << cdata.vaccinated << endl;
    x << " Vaccinated to Population Ratio : " << cdata.vacc2popratio << endl
```

```
      <<   Recovered to Confirmed Ratio :   << cdata.rec2cnfratio << endl
      << " Tested to Population Ratio : " << cdata.tst2popratio << endl;
    return x;
}

istream &operator>>(istream &x, coviddata &cdata)
{
    string statedata;
    getline(cin, statedata);
    short k = 0, count = 0;
    string first;
    while ((k = statedata.find(',')) != 0 && k <= statedata.length())
    { // comma found at position k
        first = statedata.substr(0, k);
        statedata = statedata.substr(k + 1, statedata.length() - k - 1);
        switch (count)
        {
        case 0:
        {
            cdata.state = first;
            break;
        }
        case 1:
        {
            cdata.population = mystoi(first) * 100000;
            break;
        }
        case 2:
        {
            cdata.confirmed = mystoi(first);
            break;
        }
        case 3:
        {
            cdata.active = mystoi(first);
            break;
        }
        case 4:
        {
            cdata.recovered = mystoi(first);
            break;
        }
        case 5:
        {
            cdata.deceased = mystoi(first);
            break;
        }
        case 6:
        {
            cdata.tested = 100000 * mystoi(first);
            break;
        }
```

```
            case 7:
            {
            } // return -2;}
            }; // end of inner while : extracted one field at each iteration
            count++;
        };
        if (statedata.length() > 0)
        {
            cdata.vaccinated = 100000 * mystoi(statedata); // last data value
        };
        return x;
    }
```

coviduse3.cpp

```
#include <iostream>
#include <string>
#include "coviddef3.h"
int  main(int argc, char*charv[] )
{   coviddata allcoviddata[100];
    string header;
    short statecnt = 0;
    getline(cin, header);
    while (cin >> allcoviddata[statecnt]) { ++statecnt;}
    for (int i = 0; i < statecnt; i++) { allcoviddata[i].ratios(); }
    cout << " Display covid state data with comma separated components" << endl
    for (int i = 0; i < statecnt; i++) { cout << allcoviddata[i];}
    return 0;
   }
```