```
Script started on 2022-09-26 08:56:13-05:00 [TERM="xterm-256color" TTY="/dev/pts/1'
pu06439@ares:~/CSC122/Portfolio I/Projects/Let's MAP it out$ pwd
/home/students/pu06439/CSC122/Portfolio I/Projects/Let's MAP it out
pu06439@ares:~/CSC122/Portfolio I/Projects/Let's MAP it out$ cat MapMain.info
*************************************************************************
* Upanshu Parekh                                             CSC122-001 *
*                   Project: Let's MAP it out...                        *
*                                                                       *
*   This program allows users to essentially play City Management       *
* Simulator! Users can (1) add cities with a 20-or-less character name  *
* and an x-y coordinate location, (2) calculate distances between their *
* added cities and (3) print out a list of cities that they have added. *
*                                                                       *
*   Project Base Level: 4                                               *
*   Options Selected:                                                   *
*       Level 2: Allow user to select city lists with number or name    *
*   Total Project Level: 6                                              *
*************************************************************************
pu06439@ares:~/CSC122/Portfolio I/Projects/Let's MAP it out$
pu06439@ares:~/CSC122/Portfolio I/Projects/Let's MAP it out$
pu06439@ares:~/CSC122/Portfolio I/Projects/Let's MAP it out$ show-code MapMain.cpp


MapMain.cpp:
```

```cpp
//Base Level: 4, Selected first option (LVL 2)
//Total: lvl 4+2 = 6
#include "cityManager.h"
#include <vector>
#include <string>
#include <iostream>
#include <climits>

using namespace std;
using sztp = vector<City>::size_type;

vector<City> cities{}, subCities{};
City newCity{}, city1{}, city2{};
string newName{};
double newX{0.0}, newY{0.0};
char menuSelection;

int main()
{
    cout << "\nWelcome to City Management Simulator!";
    do
    {
        cout << "\nWhat would you like to do?"
                "\n\t1) E|nter city Information"
                "\n\t2) C|alculate Distance between two cities"
                "\n\t3) P|rint All cities"
                "\n\t4) Q|uit"
                "\nYour option (capitalized letter or #): ";
        cin >> menuSelection;
        cin.ignore(INT_MAX, '\n');

        switch (menuSelection)
        {
            case 'E': case 'e': case '1':
            {
                cout << "\nYou selected [1]";
                do
                {
                    cout << "\nPlease enter the name of your new city "
                            "(Max length = " << City::MAX_CITY_NAME <<
                            "): ";
                    getline(cin, newName);
                } while (newCity.set_name(newName) == false);

                cout << "\nPlease enter x-coord of your new city: ";
                cin >> newX;
                cout << "\nPlease enter y-coord of your new city: ";
                cin >> newY;
                newCity.set_location(newX, newY);

                cities.push_back(newCity);
                cout << "\nSuccess! New city created: "
                        "\"" << newCity.get_name() << "\""
                        " located at ";
                newCity.get_location().Output();
                break;
            }
            case 'C': case 'c': case '2':
            {
                cout << "\nYou selected [2]";
                if (cities.size() < 2 )
                {
                    cout << "\nInvalid! Need 2 or more entered cities "
                            "for a distance calculation.";
                }
                else if (cities.size() == 2)
                {
                    cout << "\nDistance between "
                         << cities.at(0).get_name() << " and "
                         << cities.at(1).get_name() << " is "
                         << cities.at(0).distance(cities.at(1))
                         << ".";
                }
                else
                {
                    subCities = inputFindCities(
                        "\nPlease select the first city (name or #): ",
                        cities);
                    if (subCities.size() == 1)
                    {
                        city1 = subCities.at(0);
                    }
                    else if (subCities.size() > 1)
                    {   //               ^ Multiple cities of same name
                        cout << "\nMultiple cities of same name found! ";
                        city1 = inputFindCities(
                                "\nPlease select first city from the "
                                "list below with a numeric selection: ",
                                subCities, true).at(0);
                        //has to return a vector<City>
                        // with a single element
                    }

                    bool valid;
                    do
                    {
                        valid = true;
                        subCities = inputFindCities(
                            "\nPlease select the second city "
                            "(name or #):",
                            cities);
                        if (subCities.size() == 1)
                        {
                            city2 = subCities.at(0);
                        }
                        else if (subCities.size() > 1)
                        {   //               ^ Multiple cities of same name
                            cout << "\nMultiple cities of same name "
                                    "found! ";
                            city2 = inputFindCities(
                                    "\nPlease select second city from "
                                    "the list with a numeric selection: "
                                    , subCities, true).at(0);
                            //has to return a vector<City>
                            // with a single element
                        }
                        if (searchCityVector(city2, cities) ==
                            searchCityVector(city1, cities))
                        {
                            cout << "\nInvalid value! Second city must "
                                    "be different from first city. "
                                    "Try again.";
                            valid = false;
                        }
                    } while (valid == false);
                    cout << "\nDistance between \""
                            << city1.get_name()
                            << "\" and \""
                            << city2.get_name()
                            << "\" is "
                            << city1.distance(city2) << ".";
                }
                break;
            }
            case 'P': case 'p': case '3':
            {
                cout << "\nYou selected [3]";
                printCityVector(cities);
                break;
            }
            case 'Q': case 'q': case '4':
            {
                cout << "\nYou selected [4]";
                cout << "\nThank you for using City Management "
                        "Simulator. Goodbye!";
                break;
            }
            default:
            {
                cout << "\nInvalid selection! Try again.";
                break;
            }
        }
    } while (menuSelection != 'Q' && menuSelection != 'q' &&
             menuSelection != '4');

    return 0;
}
```

```
pu06439@ares:~/CSC122/Portfolio I/Projects/Let's MAP it out$ show-code point.h
```

point.h:

```
/*
    NOTE: I am aware that programs that are given to me in the project
    description are to be used "as is". But the remaining warnings I was
    receiving were from this "as is" class, so I had to edit it.
    I confirmed that I could edit this with you.
*/

#ifndef POINT_CLASS_HEADER_INCLUDED
#define POINT_CLASS_HEADER_INCLUDED

// A 2D point class
class Point
{
    double x, // x coordinate of point
           y; // y coordinate of point

public:

    // CPP compiler told me to use member initialization lists for these
    // 2 c'tors:
    Point(void)
        : x{0.0}, y{0.0}
    {}
    Point(double new_x, double new_y)
        : x{new_x}, y{new_y}
    {}

    // = default helped fix depecrated copy warnings
    Point(const Point & p) = default;

    void Output(void) const;   // output this point
    void Input(void);          // input this point

    // distance between this point and other
    double distance(const Point & other) const;
    // point in middle of this point and other
    Point midpoint(const Point & other) const;

    double get_x(void) const { return x; }  // accessors
    double get_y(void) const { return y; }

    void set_x(double new_x);               // mutators
    void set_y(double new_y);

    Point flip_x(void) const; // new point is this one flipped
    Point flip_y(void) const; // about specified axis

    Point shift_x(double move_by) const;  // new point is this one
    Point shift_y(double move_by) const;  // shifted move_by in the
                                          // specified direction
};

#endif
```

```
pu06439@ares:~/CSC122/Portfolio I/Projects/Let's MAP it out$ show-code point.cpp


point.cpp:
```

```
/*
    NOTE: I am aware that programs that are given to me in the project
    description are to be used "as is". But the remaining warnings I was
    receiving were from this "as is" class, so I had to edit it.
    I confirmed that I could edit this with you.
*/

#include "point.h"

#include <iostream>
#include <cmath>

using namespace std;

// read standard 2D point notation (x,y) -- ignore
// window dressing
void Point::Input(void)
{
    char dummy;
    cin >> dummy >> x >> dummy >> y >> dummy;
    return;
}

// output standard 2D point notation (x,y)
void Point::Output(void) const
{
    cout << '(' << x << ", " << y << ')';
    return;
}

// calculate distance between two 2D points --
// the one that called us and the argument
double Point::distance(const Point & other) const
{
    return sqrt(pow(x-other.x, 2.0) +
                pow(other.y-y, 2.0));
}

// calculate midpoint between two 2D points --
// the one that called us and the argument
Point Point::midpoint(const Point & other) const
{
    return Point((x+other.x)/2.0, (other.y+y)/2.0);
}

// set coordinates to programmer-specified values
void Point::set_x(double new_x)
{
    x = new_x;        // no error checking since anything is legal
    return;
}

// set coordinates to programmer-specified values
void Point::set_y(double new_y)
{
    y = new_y;        // no error checking since anything is legal
    return;
}

// in point.h the default copy c'tor was specified instead to fix
// a lot of deprecated operator= warnings

// construct Point as copy of previous point
// Point::Point(const Point & p)
// {
//     x = p.x;
//     y = p.y;
// }


// compiler told me to use member initialization lists instead here

// construct Point by default -- no values specified
// Point::Point(void)
// {
```

```
//     x = y = 0.0;
// }

// // construct Point given initial x,y values
// Point::Point(double new_x, double new_y)
// {
//     set_x(new_x);
//     set_y(new_y);
// }

// creates a point flipped about the x axis from us
Point Point::flip_x(void) const
{
    return Point(x,-y);
}

// creates a point flipped about the y axis from us
Point Point::flip_y(void) const
{
    return Point(-x,y);
}

// creates a point shifted along the x axis from us
Point Point::shift_x(double move_by) const
{
    return Point(x+move_by,y);
}

// creates a point shifted along the y axis from us
Point Point::shift_y(double move_by) const
{
    return Point(x,y+move_by);
}
```

```
pu06439@ares:~/CSC122/Portfolio I/Projects/Let's MAP it out$ show-code city.h


city.h:
```

```
#pragma once

#include "point.h"
#include <string>

class City
{
    Point location;
    std::string name;

public:

    static const std::string::size_type MAX_CITY_NAME{20};

    City(void)
        : location{},
          name{"New City"}
    {}

    City(const std::string & newName, const Point & newLoc = Point() )
        : City{}
    {
        set_name(newName);
        set_location(newLoc);
    }

    City(const std::string & newName, const double & newX,
                                      const double & newY )
        : City{}
    {
        set_name(newName);
        set_location(newX, newY);
    }

    // this was the only way I could find
    // to prevent a "implicitly-declared operator= is deprecated"
    // warning, have to use default copy c'tor
    City(const City & c) = default;

    //member functions

    inline
    double distance(const City & other) const
    { return location.distance(other.location); }

    inline
    Point get_location(void) const { return location; }

    void set_location(const Point & newLoc);

    void set_location(const double & x, const double & y);

    inline
    std::string get_name(void) const { return name; }

    bool set_name(const std::string & newName);
};
```

```
pu06439@ares:~/CSC122/Portfolio I/Projects/Let's MAP it out$ show-code city.cpp


city.cpp:
```

```
#include "city.h"

void City::set_location(const Point & newLoc)
{
    location.set_x(newLoc.get_x());
    location.set_y(newLoc.get_y());
}

void City::set_location(const double & x, const double & y)
{
    location.set_x(x);
    location.set_y(y);
}

bool City::set_name(const std::string & newName)
{
    bool okay{false};
    if (newName.size() <= MAX_CITY_NAME)
    {
        name = newName;
        okay = true;
    }
    return okay;
}
```

```
pu06439@ares:~/CSC122/Portfolio I/Projects/Let's MAP it out$ show-code cityManager
```

cityManager.h:

```
#pragma once

#include "city.h"
#include <vector>
#include <string>

void printCityVector (const std::vector<City> & cityVector);

std::vector<City>::size_type
searchCityVector (const std::string & cityName,
                  const std::vector<City> & cityVector,
                  const std::vector<City>::size_type & start = 0);

std::vector<City>::size_type
searchCityVector (const City & city,
                  const std::vector<City> & cityVector,
                  const std::vector<City>::size_type & start = 0);

std::vector<City>
inputFindCities(const std::string & prompt,
                const std::vector<City> & cityVector,
                const bool & numericSel = false);
```

```
pu06439@ares:~/CSC122/Portfolio I/Projects/Let's MAP it out$ show-code cityManager
```

cityManager.cpp:

```
#include "cityManager.h"
#include <vector>
#include <string>
#include <iostream>
#include <cstdlib>
#include <cmath>

using namespace std;
using sztp = vector<City>::size_type;

void printCityVector (const vector<City> & cityVector)
{
    sztp vecSize{cityVector.size()};
    cout << "\nCity List: ";
    for (sztp i{0}; i != vecSize; ++i)
    {
        cout << "\n\t[" << i + 1 << "] \"" << cityVector.at(i).get_name()
        << "\" located at ";
        cityVector.at(i).get_location().Output();
    }
    cout << '\n';
    return;
}

sztp searchCityVector (const string & cityName,
                       const vector<City> & cityVector,
                       const sztp & start)
{
    sztp size{cityVector.size()};
    for (sztp i = start; i != size; ++i)
    {
        if (cityVector.at(i).get_name() == cityName)
        {
            return i;
        }
    }
    return -1;
}

sztp
searchCityVector (const City & city,
                  const vector<City> & cityVector,
                  const sztp & start)
{
    string cityName{city.get_name()},
           cVecName{};
    double cityX{city.get_location().get_x()},
           cityY{city.get_location().get_y()},
           cVecX, cVecY;
    sztp size{cityVector.size()};
    for (sztp i = start; i != size; ++i)
    {
        cVecName = cityVector.at(i).get_name();
        cVecX = cityVector.at(i).get_location().get_x();
        cVecY = cityVector.at(i).get_location().get_y();

        if (cVecName == cityName &&
            fabs(cityX - cVecX) <= 1e-6 && fabs(cityY - cVecY) <= 1e-6)
        {
            return i;
        }
    }
    return -1;
}

//prompts the user for city selection, finds cities in main
//vector that have the same name as user input
//detects if user wishes to input just a position from the list
//instead of a name
vector<City> inputFindCities(const string & prompt,
                             const vector<City> & cityVector,
                             const bool & numericSel)
{                               // ^ default value = false
    vector<City> subCities{};
    string input{};
    sztp cityIndex{0}, atollResult;

    printCityVector(cityVector);
    do
    {
        cout << prompt;
        getline(cin, input);

        //atoll() returns a long long int, sztp is usually unsigned
        //long long int, both are 64 bits usually
        //if atoll() returns negative, index will underflow, but it is
        //an invalid position 99% of the time with the sztp datatype
        atollResult = atoll(input.c_str());
        cityIndex = atollResult - 1;

        if (numericSel == true || atollResult != 0)
        {
            if (cityIndex >= cityVector.size())
            {
                cout << "\nInvalid selection! Try again.";
            }
            else if (cityIndex < cityVector.size())
            {
                subCities.push_back(cityVector.at(cityIndex));
                return subCities;
            }
        }
    } while (cityIndex >= cityVector.size() && numericSel == true);
    //if numeric selection requested and invalid conversion, then ask
    //user again until it is a valid value for cityIndex (> 1)

    do
    {
        cityIndex = 0;
        do
        {
            cityIndex = searchCityVector(input, cityVector, cityIndex);
            if (cityIndex != static_cast<sztp>(-1))
            {            //   ^ CPP was complaining about old-style casts
                subCities.push_back(cityVector.at(cityIndex));
                ++cityIndex;
            }
        } while (cityIndex != static_cast<sztp>(-1));
                        //    ^ CPP was complaining about old-style casts
        if (subCities.size() == 0)
        {
            cout << "\nNo city with that name found! Try again.";
            cout << prompt;
            getline(cin, input);
        }
    } while (subCities.size() == 0);
    //if input string is valid num -> stored in cityIndex

    return subCities;
}
```

```
pu06439@ares:~/CSC122/Portfolio I/Projects/Let's MAP it out$ CPP MapMain.cpp point
MapMain.cpp***
city.cpp...
cityManager.cpp...
point.cpp...


pu06439@ares:~/CSC122/Portfolio I/Projects/Let's MAP it out$ ./MapMain.out

Welcome to City Management Simulator!
What would you like to do?
        1) E|nter city Information
        2) C|alculate Distance between two cities
        3) P|rint All cities
        4) Q|uit
Your option (capitalized letter or #): E

You selected [1]
Please enter the name of your new city (Max length = 20): CanIReallyEnterACityNameI

Please enter the name of your new city (Max length = 20): Chicago

Please enter x-coord of your new city: 1.5

Please enter y-coord of your new city: 2.5

Success! New city created: "Chicago" located at (1.5, 2.5)
What would you like to do?
        1) E|nter city Information
        2) C|alculate Distance between two cities
        3) P|rint All cities
        4) Q|uit
Your option (capitalized letter or #): 2
```

```
You selected [2]
Invalid! Need 2 or more entered cities for a distance calculation.
What would you like to do?
        1) E|nter city Information
        2) C|alculate Distance between two cities
        3) P|rint All cities
        4) Q|uit
Your option (capitalized letter or #): e

You selected [1]
Please enter the name of your new city (Max length = 20): New York

Please enter x-coord of your new city: 25.8

Please enter y-coord of your new city: 13.2

Success! New city created: "New York" located at (25.8, 13.2)
What would you like to do?
        1) E|nter city Information
        2) C|alculate Distance between two cities
        3) P|rint All cities
        4) Q|uit
Your option (capitalized letter or #): C

You selected [2]
Distance between Chicago and New York is 26.5515.
What would you like to do?
        1) E|nter city Information
        2) C|alculate Distance between two cities
        3) P|rint All cities
        4) Q|uit
Your option (capitalized letter or #): p

You selected [3]
City List:
        [1] "Chicago" located at (1.5, 2.5)
        [2] "New York" located at (25.8, 13.2)

What would you like to do?
        1) E|nter city Information
        2) C|alculate Distance between two cities
        3) P|rint All cities
        4) Q|uit
Your option (capitalized letter or #): 1

You selected [1]
Please enter the name of your new city (Max length = 20): New York

Please enter x-coord of your new city: 34.2

Please enter y-coord of your new city: 37.4

Success! New city created: "New York" located at (34.2, 37.4)
What would you like to do?
```

```
        1) E|nter city Information
        2) C|alculate Distance between two cities
        3) P|rint All cities
        4) Q|uit
Your option (capitalized letter or #): 1

You selected [1]
Please enter the name of your new city (Max length = 20): Atlantis

Please enter x-coord of your new city: -834.25

Please enter y-coord of your new city: 93.58

Success! New city created: "Atlantis" located at (-834.25, 93.58)
What would you like to do?
        1) E|nter city Information
        2) C|alculate Distance between two cities
        3) P|rint All cities
        4) Q|uit
Your option (capitalized letter or #): 2

You selected [2]
City List:
        [1] "Chicago" located at (1.5, 2.5)
        [2] "New York" located at (25.8, 13.2)
        [3] "New York" located at (34.2, 37.4)
        [4] "Atlantis" located at (-834.25, 93.58)

Please select the first city (name or #): New York

Multiple cities of same name found!
City List:
        [1] "New York" located at (25.8, 13.2)
        [2] "New York" located at (34.2, 37.4)

Please select first city from the list below with a numeric selection: 9

Invalid selection! Try again.
Please select first city from the list below with a numeric selection: 2

City List:
        [1] "Chicago" located at (1.5, 2.5)
        [2] "New York" located at (25.8, 13.2)
        [3] "New York" located at (34.2, 37.4)
        [4] "Atlantis" located at (-834.25, 93.58)

Please select the second city (name or #):3

Invalid value! Second city must be different from first city. Try again.
City List:
        [1] "Chicago" located at (1.5, 2.5)
        [2] "New York" located at (25.8, 13.2)
        [3] "New York" located at (34.2, 37.4)
        [4] "Atlantis" located at (-834.25, 93.58)
```

```
Please select the second city (name or #):2

Distance between "New York" and "New York" is 25.6164.
What would you like to do?
        1) E|nter city Information
        2) C|alculate Distance between two cities
        3) P|rint All cities
        4) Q|uit
Your option (capitalized letter or #): c

You selected [2]
City List:
        [1] "Chicago" located at (1.5, 2.5)
        [2] "New York" located at (25.8, 13.2)
        [3] "New York" located at (34.2, 37.4)
        [4] "Atlantis" located at (-834.25, 93.58)

Please select the first city (name or #): 4

City List:
        [1] "Chicago" located at (1.5, 2.5)
        [2] "New York" located at (25.8, 13.2)
        [3] "New York" located at (34.2, 37.4)
        [4] "Atlantis" located at (-834.25, 93.58)

Please select the second city (name or #):New York

Multiple cities of same name found!
City List:
        [1] "New York" located at (25.8, 13.2)
        [2] "New York" located at (34.2, 37.4)

Please select second city from the list with a numeric selection: 1

Distance between "Atlantis" and "New York" is 863.798.
What would you like to do?
        1) E|nter city Information
        2) C|alculate Distance between two cities
        3) P|rint All cities
        4) Q|uit
Your option (capitalized letter or #): 2

You selected [2]
City List:
        [1] "Chicago" located at (1.5, 2.5)
        [2] "New York" located at (25.8, 13.2)
        [3] "New York" located at (34.2, 37.4)
        [4] "Atlantis" located at (-834.25, 93.58)

Please select the first city (name or #): 321321

Invalid selection! Try again.
No city with that name found! Try again.
```

```
Please select the first city (name or #): 1

No city with that name found! Try again.
Please select the first city (name or #): Chicago

City List:
        [1] "Chicago" located at (1.5, 2.5)
        [2] "New York" located at (25.8, 13.2)
        [3] "New York" located at (34.2, 37.4)
        [4] "Atlantis" located at (-834.25, 93.58)

Please select the second city (name or #):4

Distance between "Chicago" and "Atlantis" is 840.698.
What would you like to do?
        1) E|nter city Information
        2) C|alculate Distance between two cities
        3) P|rint All cities
        4) Q|uit
Your option (capitalized letter or #): P

You selected [3]
City List:
        [1] "Chicago" located at (1.5, 2.5)
        [2] "New York" located at (25.8, 13.2)
        [3] "New York" located at (34.2, 37.4)
        [4] "Atlantis" located at (-834.25, 93.58)

What would you like to do?
        1) E|nter city Information
        2) C|alculate Distance between two cities
        3) P|rint All cities
        4) Q|uit
Your option (capitalized letter or #): 3

You selected [3]
City List:
        [1] "Chicago" located at (1.5, 2.5)
        [2] "New York" located at (25.8, 13.2)
        [3] "New York" located at (34.2, 37.4)
        [4] "Atlantis" located at (-834.25, 93.58)

What would you like to do?
        1) E|nter city Information
        2) C|alculate Distance between two cities
        3) P|rint All cities
        4) Q|uit
Your option (capitalized letter or #): Q

You selected [4]
Thank you for using City Management Simulator. Goodbye!pu06439@ares:~/CSC122/Portf
pu06439@ares:~/CSC122/Portfolio I/Projects/Let's MAP it out$ exit
exit
```

```
Script done on 2022-09-26 09:06:19-05:00 [COMMAND_EXIT_CODE="0"]
```