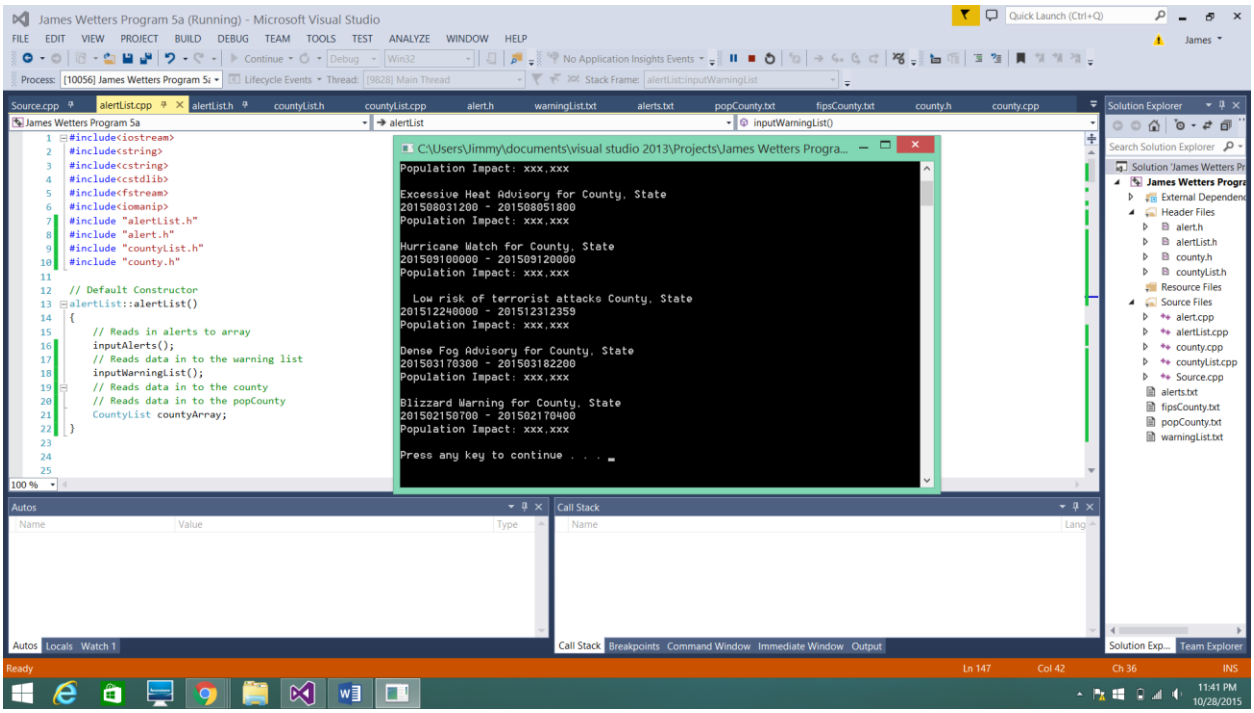
Program 5



Out Put

Winter Storm Warning for County, State

201502121300 - 201502131200

Population Impact: xxx,xxx

Significant risk of terrorist attacks County, State

201507010000 - 201507112359

Population Impact: xxx,xxx

Excessive Heat Advisory for County, State

201508031200 - 201508051800

```
Population Impact: xxx,xxx


Hurricane Watch for County, State
201509100000 - 201509120000
Population Impact: xxx,xxx


  Low risk of terrorist attacks County, State
201512240000 - 201512312359
Population Impact: xxx,xxx


Dense Fog Advisory for County, State
201503170300 - 201503182200
Population Impact: xxx,xxx


Blizzard Warning for County, State
201502150700 - 201502170400
Population Impact: xxx,xxx


Press any key to continue . . .
```

Source

```
// Program 5 description
// Author James Wetters

#include<iostream>
#include<string>
#include<cstring>
#include<cstdlib>
#include<fstream>
#include<iomanip>
#include "alert.h"
#include "alertList.h"

using namespace std;

// Constant Variables
```

```
// Prototypes


int main()
{
	// Initilize variables
	int goodData = 0;

	// Initilize objects
	alertList a;

	// Print Alerts
	//b.inputCountyFips();
	a.print();

	// End Of Program
	system("pause");
	return 0;
}

// Functions-----------------------------------------------------------------
```

Alertlist.cpp

```
#include<iostream>
#include<string>
#include<cstring>
#include<cstdlib>
#include<fstream>
#include<iomanip>
#include "alertList.h"
#include "alert.h"
#include "countyList.h"
#include "county.h"

// Default Constructor
alertList::alertList()
{
	// Reads in alerts to array
	inputAlerts();
	// Reads data in to the warning list
	inputWarningList();
	// Reads data in to the county
	// Reads data in to the popCounty
	CountyList countyArray;
}



//*********************************************************************
//	Input File
```

```
//
// Pre: Objects come in as an array, alerts come in as a text file
// Post:
//*************************************************************************

// Input Alerts
void alertList::inputAlerts()
{
	// Open File
	ifstream inputFile("alerts.txt");

	// Test File
	if (inputFile.fail())
	{
		cout << "Problem opening file";
		system("pause");
		exit(-1);
	}

	// variables
	string temp;

	// Priming read
	getline(inputFile, temp, ',');

	// Read in data file
	while (!inputFile.eof())
	{
		// Firts Read
		alerts[numElems].setCountyFipsCode(temp);

		// Read in secound line
		getline(inputFile, temp, ',');
		alerts[numElems].setStartDateAndTime(temp);

		// Read in third line
		getline(inputFile, temp, ',');
		alerts[numElems].setEndDateAndTime(temp);

		// Read in forth line
		getline(inputFile, temp);
		alerts[numElems].setWarningCode(temp);

		numElems++;			// Increase goodData by 1

		// Read in first line
		getline(inputFile, temp, ',');

	}

	// Close file
	inputFile.close();
}

//*************************************************************************
//	Input Warning List File
//
// Pre: Objects come in as an array, alerts come in as a text file
```

```
// Post:
//*****************************************************************************

// Input Warnings
void alertList::inputWarningList()
{
	// Open File
	ifstream inputFile("warningList.txt");

	// Test File
	if (inputFile.fail())
	{
		cout << "Problem opening file";
		system("pause");
		exit(-1);
	}

	// variables
	string temp;

	// skip 3 lines
	getline(inputFile, temp);
	getline(inputFile, temp);
	getline(inputFile, temp);

	// Read in weather warnings
	for (int i = 0; i < 23; i++)
	{
		// Initilize variables
		string subTemp;

		// Substring the last 2 characters
		getline(inputFile, temp, ' ');
		// Assign characters to a temp string
		subTemp = temp.substr(1, 2);
		// Assign string to the warning list
		weatherWarningList[0][i] = subTemp;

		// Get next line for processing
		getline(inputFile, temp);
		weatherWarningList[1][i] = temp;

	}

	// Skip 8 lines
	for (int i = 0; i < 8; i++)
	{
		getline(inputFile, temp);
	}

	// Read in security warning
	for (int i = 0; i < 5; i++)
	{
		// Read until space
		getline(inputFile, temp, ' ');
		// Assign color code to col 0
		nationalWarningList[0][i] = temp;
```

```
                // Read until end of the line
                getline(inputFile, temp);
                // Assign warning message to col 1
                nationalWarningList[1][i] = temp;
        }

        // Close file
        inputFile.close();
}


//*************************************************************************
//      Sort
//
// Pre: Objects come in as an array
// Post: Objects sorted based on security
// warnings first than weather warnings in the order of severity; warnings,
// watches, and advisories. Objects leave sorted based on serverity.
//*************************************************************************

void alertList::sort()
{

}


//*************************************************************************
//      Print
//
// Pre:
// Post:
//*************************************************************************

void alertList::print()
{
        // Write each warning
        for (int i = 0; i < numElems; i++)
        {
                // Print Warning
                warning(alerts[i].getWarningCode());

                // Print City and state
                cout << "County, State" << endl;

                // Print start date
                cout << alerts[i].getStartDateAndTime() << " - ";

                // Print end date
                cout << alerts[i].getEndDateAndTime() << endl;

                // Print population impact
                cout << "Population Impact: xxx,xxx" << "\n" << endl;


        }
}

//*************************************************************************
```

```
//     Warning
//
// Pre: waring string comes in
// Post: warning writen to console comes out
//*************************************************************************
void alertList::warning(string test)
{
	// Initilize variables
	string temp;
	bool found = false;
	int search = 0;

	// Test for security warning
	for (int i = 0; i < 5; i++)
	{
		if (test == nationalWarningList[0][i])
		{
			cout << nationalWarningList[1][i] << " ";
			found = true;
		}

	}

	// If found equals true then leave the fuction
	// otherwise find the weather advisery and severity
	if (found)
	{
		return;
	}
	else
	{
		//Initilize
		string compare;
		compare = test.substr(1, 2);
		// Test for weather warning
		for (int search = 0; search < 23; search++)
		{
			if (compare == weatherWarningList[0][search])
			{
				cout << weatherWarningList[1][search] << " ";
			}
		}

		compare = test.substr(0, 1);
		string W = "W", A = "A", Y = "Y";

		// Find severity
		if (compare == W)
		{
			cout << "Warning for ";
		}
		else if (compare == A)
		{
			cout << "Watch for ";
		}
		else
		{
			cout << "Advisory for ";
```

```
                }
        }
}

//*************************************************************************
//      Format Date
//
// Pre:
// Post:
//*************************************************************************

string alertListdate(string test)
{
        return test;
}


//*************************************************************************
//      Write County and State
//
// Pre:
// Post:
//*************************************************************************

string alertList::writeCountyState(string test)
{
        return test;
}

//*************************************************************************
//      Write County and State
//
// Pre:
// Post:
//*************************************************************************

string alertList::writePopulation(string test)
{
        return test;
}
```

alertList.h

```
// Alert Header File
// Author James Wetters

// Includes
#ifndef alertListInfo_H
#define alertListInfo_H

#include <iostream>
#include <iomanip>
#include <string>
```

```
#include <fstream>
#include <cstdlib>
#include <cstring>
#include "alert.h"
#include "county.h"
#include "countyList.h"

// Name Space
using namespace std;

// Constants
const int ALERTMAXARRAY = 10;
const int MAXWEATHERARRAYCOL = 2;
const int MAXWEATHERARRAYROW = 25;
const int MAXNATIONALARRYCOL = 2;
const int MAXNATIONALARRYCROW = 10;

class alertList
{

private:
	// Data members
	int numElems = 0;

	// Array of Alerts
	string weatherWarningList[MAXWEATHERARRAYCOL][MAXWEATHERARRAYROW];
	string nationalWarningList[MAXNATIONALARRYCOL][MAXNATIONALARRYCROW];
	alert alerts[ALERTMAXARRAY];

public:
	// Constructors
	alertList();

	// Sets


	// Gets


	// Functions Prototype
	void inputAlerts();
	void inputWarningList();
	void sort();
	void print();
	void warning(string test);
	string date(string test);
	string writeCountyState(string test);
	string writePopulation(string test);

};
#endif
```

Countylist.cpp

```
// CountyList Source File
```

```cpp
// Author James Wetters

#include <iostream>
#include <iomanip>
#include <string>
#include <fstream>
#include <cstdlib>
#include <cstring>
#include "countyList.h"

// Namespace
using namespace std;


CountyList::CountyList()
{
	inputCountyFips();
	inputCounty();
}

//*************************************************************************
//	Input Fips and Pop
//
// Pre:
// Post:
//*************************************************************************

void CountyList::inputCountyFips()
{
	ifstream inputFile("popCounty.txt");

	// Test File
	if (inputFile.fail())
	{
		cout << "Problem opening file";
		system("pause");
		exit(-1);
	}

	// variables
	string temp;
	numElems = 0;
	// Prime
	getline(inputFile, temp, ',');

	while (!inputFile.eof())
	{
		// Set fips
		allCounties[numElems].setFips(temp);

		// Get population and set it
		getline(inputFile, temp);
		allCounties[numElems].setPopulation(temp);

		// Get next fips
		getline(inputFile, temp, ',');
		numElems++;
	}
```

```
	inputFile.close();
}

//*************************************************************************
//	Input County
//
// Pre:
// Post:
//*************************************************************************

void CountyList::inputCounty()
{
	// Input County Name
	ifstream inputFile("fipsCounty.txt");

	// Test File
	if (inputFile.fail())
	{
		cout << "Problem opening file";
		system("pause");
		exit(-1);
	}

	// Initilize variables
	string temp;
	int countyNum = 0;

	// Prime
	getline(inputFile, temp, ' ');

	// Read in fips codes check and set countys
	while (!inputFile.eof())
	{
		bool found = false;
		int count = 0;
		// Set fips
		// look for fips code
		while (!found && count < numElems)
		{
			// test for fips code
			if (temp == allCounties[count].getFips())
			{
				// If fips code found set county name
				getline(inputFile, temp);
				allCounties[countyNum].setCounty(temp);

				// If found set to true
				found = true;
			}

			// Check next fips code
			count++;
		}

		// If not found get next line
		if (found == false)
		{
```

```
                        getline(inputFile, temp);
                }

                // Get next fips
                getline(inputFile, temp, ' ');

                // Reset
                count = 0;
                found = false;
        }

        inputFile.close();
}
```

countyList.h

```
// County Header File
// Author James Wetters

// Includes
#ifndef countyListInfo_H
#define countyListInfo_H

#include <iostream>
#include <iomanip>
#include <string>
#include <fstream>
#include <cstdlib>
#include <cstring>
#include "county.h"

// Name Space
using namespace std;

// Constants
const int MAXCOUNTYARRAY = 3300;


class CountyList
{

private:
        // Data members
        County allCounties[MAXCOUNTYARRAY];
        int numElems;

public:
        // Constructors
        CountyList();

        // Sets
        void setNumElems(int change)
        {
                numElems = change;
```

```
	}

	// Gets
	int getNumElems() const
	{
		return numElems;
	}

	// Functions Prototype
	void inputCountyFips();
	void inputCounty();

};
#endif
```

Alert.h

```
// Alert Header File
// Author James Wetters

// Includes
#ifndef alertInfo_H
#define alertInfo_H

#include <iostream>
#include <iomanip>
#include <string>
#include <fstream>
#include <cstdlib>
#include <cstring>

// Name Space
using namespace std;


class alert
{
private:
	// Data members
	string countyFipsCode;
	string startDateAndTime;
	string endDateAndTime;
	string warningCode;


public:
	alert();

	// Sets
	void setCountyFipsCode(string change)
	{
		countyFipsCode = change;
	}
	void setStartDateAndTime(string change)
	{
```

```
		startDateAndTime = change;
	}
	void setEndDateAndTime(string change)
	{
		endDateAndTime = change;
	}
	void setWarningCode(string change)
	{
		warningCode = change;
	}

	// Gets
	string getCountyFipsCode() const
	{
		return countyFipsCode;
	}
	string getStartDateAndTime() const
	{
		return startDateAndTime;
	}
	string getEndDateAndTime() const
	{
		return endDateAndTime;
	}
	string getWarningCode() const
	{
		return warningCode;
	}


	// Functions Prototype
	void print();

};
#endif
```

Alert.cpp

```
#include<iostream>
#include<string>
#include<cstring>
#include<cstdlib>
#include<fstream>
#include<iomanip>
#include "alert.h"

// Default Constructor
alert::alert()
{
}

//*************************************************************************
//	Print
//
//
```

```
//************************************************************************

void alert::print()
{
	cout << warningCode << " " << countyFipsCode << endl;
	cout << startDateAndTime << " - " << endDateAndTime << endl;
	cout << "Population Impact: " << endl;
}
```

County.cpp

```
// County Header File
// Author James Wetters

#include <iostream>
#include <iomanip>
#include <string>
#include <fstream>
#include <cstdlib>
#include <cstring>
#include "county.h"

// Namespace
using namespace std;

County::County()
{}
```

County.h

```
// County Header File
// Author James Wetters

// Includes
#ifndef countyInfo_H
#define countyInfo_H

#include <iostream>
#include <iomanip>
#include <string>
#include <fstream>
#include <cstdlib>
#include <cstring>

// Name Space
using namespace std;

class County
```

```
{

private:
	// Data members
	string fips;
	string population;
	string county;

public:
	// Constructor
	County();

	// Sets
	void setFips(string change)
	{
		fips = change;
	}
	void setPopulation(string change)
	{
		population = change;
	}
	void setCounty(string change)
	{
		county = change;
	}

	// Gets
	string getFips() const
	{
		return fips;
	}
	string getPopulation() const
	{
		return population;
	}
	string getCounty() const
	{
		return county;
	}

	// Functions Prototype

};
#endif
```