Solar Panel Efficiency

New Mexico Supercomputing Challenge

Final Report

April 3rd, 2013

SCC Team 4

Albuquerque Institute for Math and Science

Team Members:

Charlie Rhykerd, Frankie Hendrick, Jared Kileen

Executive Summary

The current energy trend in the United States is unsustainable. Many alternative energy solutions, such as solar, wind, and geothermal energies have been examined but never implemented on a large scale. The goal of this project was to find out how cost-effective a large-scale solar implementation would be. The project has not reached a conclusive stage as of yet, but the program has determined the most effective states for a solar implementation. The code we wrote takes user input, specifically the state of the installation and the number of panels used then takes that input, finds the correct number of hours of sunlight per year for the state, which is next multiplied by the number of panels used, and finally multiplies by the power produced per panel, outputting power produced per year in kilowatt-hours.

Problem Statement

In America, most of our energy comes from fossil fuels, and fossil fuels are non-renewable. A possible way to remedy this problem would be to use a form of renewable energy, such as solar power. Our project seeks to determine how much power an implementation of solar power would produce.

Method

To solve the problem, the team developed a C++ program to determine the power output of a number of solar panels over one year in a given state in the U.S. The code begins by defining the variables:

```
        int numberOfPanels, powerProduced;
    string initialsInput;
        double panelOutput = (6.02/24); //Power per day divided by 24 yields power per hour
        int arrayPosition = -1;
        int sunshineAmount;
string initialsArray [] = {"AL", "AK", "AZ", "AR", "CA", "CO", "CT", "DE", "FL", "GA", "HI", "ID", "IL", "IN", "IA", "KS", "KY", "LA", "ME", "MD", "MA", "MI", "MN", "MS", "MO", "MT", "NE", "NV", "NH", "NJ", "NM", "NY", "NC", "ND", "OH", "OK", "OR", "PA", "RI", "SC", "SD", "TN", "TX", "UT", "VT", "VA", "WA", "WV", "WI", "WY"};

        int sunshineArray[] = {2641,2061,3806,2771,3055,3204,2585,2204,2927,2986,3022,2993,2567,2440,2691,2922,2514,3649,2513,2582,2634,2392,2711,2720,2690,2698,2672,3646,2519,2499,3415,2120,2651,2738,2183,3089,2341,2614,2606,2826,2947,2510,2850,3029,2295,2829,2170,1890,2428,3073};
```

The code defines Number of Panels, Power Produced, Array Position, and Sunshine Amount as integers, Initials Input as a string, and Panel Output as a double. It also defines Array Position's value as -1. Next, the program defines the arrays: Initials Array and Sunshine Array. The program then asks the user to input the number of panels used (Number of Panels) and the state of installation in the form of the state's abbreviation (Initials Input.) The code then uses a simple search function:

```
int counter = 0;
   while (counter < 50)
        {
       if (initialsInput == initialsArray[counter])
                {
          arrayPosition = counter;
                }
       counter++;
        }
```

This function looks through the array Initials Array and looks at each value in turn, stopping when it finds the initials that match the Initials Input. If no match is found, the program goes to the error function:

```
if (arrayPosition == -1)
        {
       cout << "There was an error."
       << "\n\"" << initialsInput << "\" was not a valid input; the program will break.";
       exit(1);
        }
```

This function checks to see if the user's input is valid. At the beginning of the program, Array Position was defined as -1. If the state initials input by the user are invalid, the search function will not change the value of Array Position that was set at the beginning. The error function is initiated only if Array Position is equal to -1. If the value of Array Position was set to something other than -1 by the search function, the program goes on to perform the calculation:

```
sunshineAmount = sunshineArray[arrayPosition];
   powerProduced = sunshineAmount * numberOfPanels * panelOutput;
```

This function uses the user's input to find the power produced by the panels. It uses Array Position to find the number of hours of sunlight per year by retrieving the value stored in Sunshine Array, and that is then multiplied by the number of panels and then by the output per panel. It then puts that into the output function:

```
    cout << "The power produced in the state " << initialsInput <<  " was " <<
powerProduced;
```

The output function tells the user the state they entered, and how much power was produced.

Results

Our data is not complete, as we do not have the costs of solar implementation, but our research has determined the best states for solar panels.

State	Sun hours per year	State	Sun hours per year
AZ	3806	NE	2672
LA	3649	NC	2651
NV	3646	AL	2641
NM	3415	MA	2634
CO	3204	PA	2614
OK	3089	RI	2606
WY	3073	CT	2585
CA	3055	MD	2582
UT	3029	IL	2567
HI	3022	NH	2519
ID	2993	KY	2514
GA	2986	ME	2513
SD	2947	TN	2510
FL	2927	NJ	2499
KS	2922	IN	2440
TX	2850	WI	2428
SC	2826	MI	2392
VA	2829	OR	2341
AR	2771	VT	2295
ND	2738	DE	2204
MS	2720	OH	2183
MN	2711	WA	2170
MT	2698	NY	2120
IA	2691	AK	2061
MO	2690	WV	1890

This table shows the hours of sun each year for each state in the U.S, listed from greatest to least sun hours.

Conclusion

We concluded that New Mexico is one of the best places for solar panel implementation. We have the fifth highest sun hours per year in the U.S, and we have lots of flat ground to install panels. Other states that may benefit from large-scale solar implementation include Arizona, Nevada, and Colorado. All of these states have similar features: some large flat areas, and some of the highest yearly sun hours.

Acknowledgements

We would like to thank Nico, Walid, Israel, RJ, and Roderick, for helping us with the code and helping us learn C++.

We would also like to thank Ms. Fey, Mr. Harris, Mr. Watje, Mr. Hostetler, Mr. Frazier, Mr. Tyler, and everybody who helped with the Supercomputing Challenge.

Our most significant achievement was learning C++ and learning how to work as a team. We learned a lot about leadership and about programming.

Code

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

using namespace std;

int main ()
{
	//definitions
	int numberOfPanels, powerProduced;
   string initialsInput;
	double panelOutput = (6.02/24); //Power per day divided by 24 yields power per hour
	int arrayPosition = -1;
	int sunshineAmount;

	//definitions of Arrays

	string initialsArray [] = {"AL", "AK", "AZ", "AR", "CA", "CO", "CT", "DE", "FL", "GA", "HI", "ID", "IL", "IN", "IA", "KS", "KY", "LA", "ME", "MD", "MA", "MI", "MN", "MS", "MO", "MT", "NE", "NV", "NH", "NJ", "NM", "NY", "NC", "ND", "OH", "OK", "OR", "PA", "RI", "SC", "SD", "TN", "TX", "UT", "VT", "VA", "WA", "WV", "WI", "WY"};

	int sunshineArray[] = {2641,2061,3806,2771,3055,3204,2585,2204,2927,2986,3022,2993,2567,2440,2691,2922,2514,3649,2513,2582,2634,2392,2711,2720,2690,2698,2672,3646,2519,2499,3415,2120,2651,2738,2183,3089,2341,2614,2606,2826,2947,2510,2850,3029,2295,2829,2170,1890,2428,3073};

   //user input
   //ask user for number of panels
   cout << "Type number of panels: ";
   cin >> numberOfPanels;

   //ask for state initials
   cout << "Type state initials: ";
   cin >> initialsInput;

   //searching
   int counter = 0;
   while (counter < 50)
	{
```

```
        if (initialsInput == initialsArray[counter])
                {
           arrayPosition = counter;
                }
        counter++;
         }

    //error-checking
    if (arrayPosition == -1)
         {
        cout << "There was an error."
        << "\n\"" << initialsInput << "\" was not a valid input; the program will break.";
        exit(1);
         }

    //calculations
    sunshineAmount = sunshineArray[arrayPosition];
    powerProduced = sunshineAmount * numberOfPanels * panelOutput;

    //report and end
    cout << "The power produced in the state " << initialsInput <<  " was " <<
powerProduced;
    return 0;
}
```