Assignment 4

Kevin Nolan
CS 3305 – Section 01

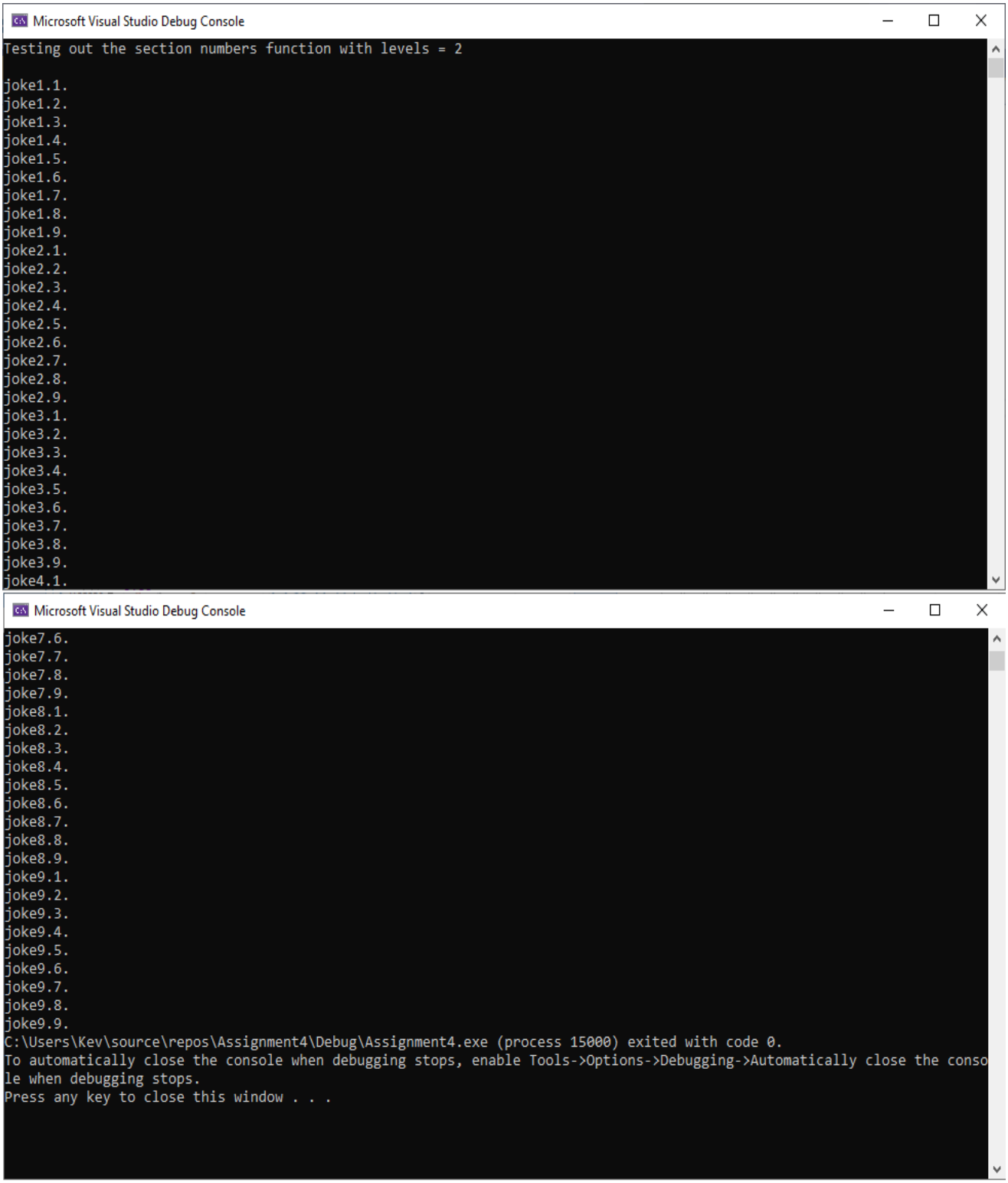
Description: This assignment implements 2 programming projects part 1 and 2. The first part prints out section numbers of a prefix value by recursively calling itself (Assignment4a.cpp). The second part prints out a * pattern based on the values provided by recursively calling itself twice(Assignment4b.cpp).

Assignment 4a (Part 1) - Source Code:

```
// Assignment 4 - Part 1
// by: Kevin Nolan
//
// This program implements part 1 of assignment 4. It takes a prefix string value
// and uses recursion to print out "section" numbers to the end of each string.

#include <iostream>
#include <string>

void numbers(std::ostream& outs, const std::string& prefix, unsigned int levels);

int main()
{
    std::cout << "Testing out the levels function!!! \n \n";
    std::ostream& os = std::cout;

    numbers(os, "joke", 2);

}

//Function to print out section numbers of prefix value
void numbers(std::ostream& outs, const std::string& prefix, unsigned int levels)
{
      std::string newString = prefix;

      if (levels == 0)
            outs << prefix;
      else
      {
            for (int j = 49; j < 58; j++) //uses char ascii values for 1-9
            {
                  numbers(outs, newString + (char)j + '.' , levels - 1);

                  if(j != 57 ) //Prints out new line unless it's reached the end!
                      std::cout << "\n";
            }

      }

 }
```

Program Output - Screenshot

Assignment4a.cpp running with levels =2 (not all #s shown)

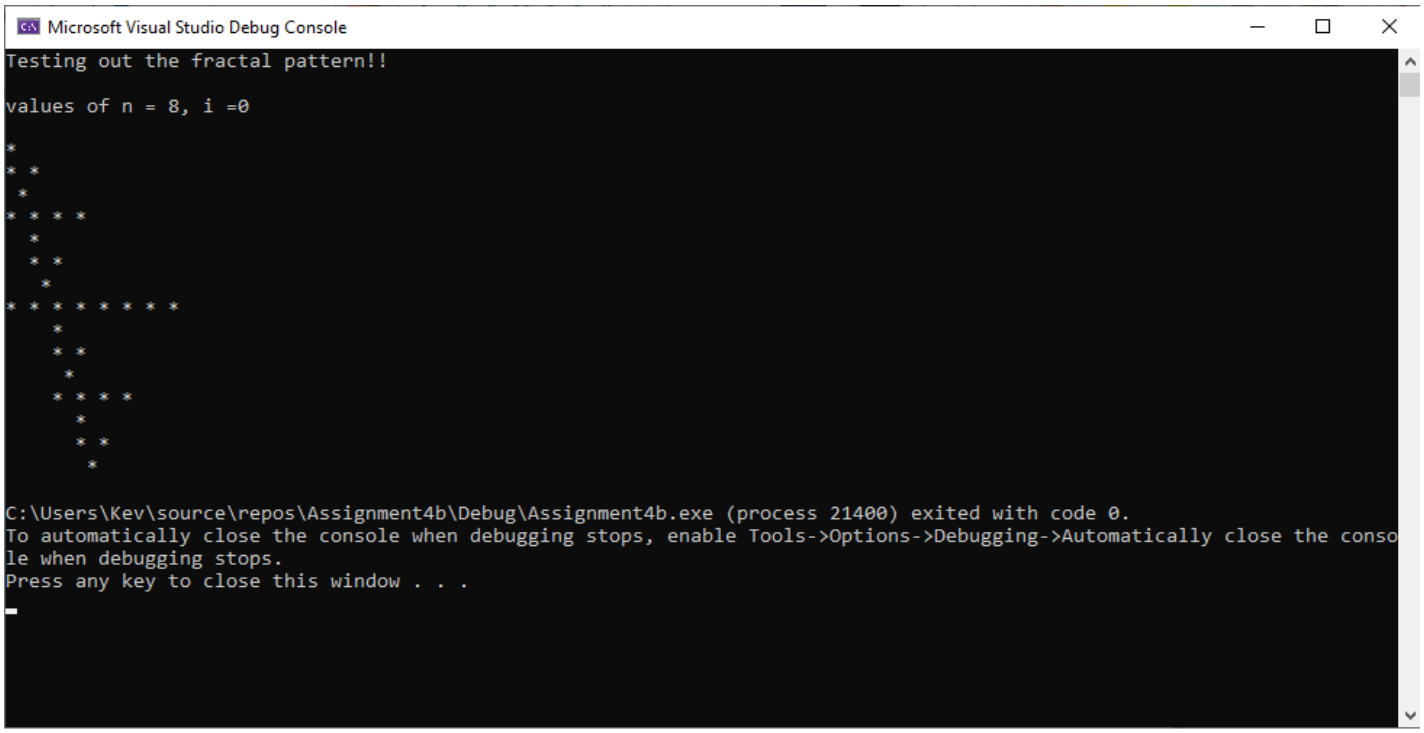


Assignment 4b (Part 2) - Source Code:

```
// Assignment 4 - Part 2
// by: Kevin Nolan
//
// This program implements part 2 of assignment 4. It takes a value of power of 2 and
// index i and prints the *** pattern based on the inputs given using recursion. In
// this case it uses 2 recursive calls in order to create the appropriate display.

#include <iostream>

using namespace std;

void pattern(ostream& outs, unsigned int n, unsigned int i);

int main()
{
   cout << "Testing out the fractal pattern!! \n \n" << "values of n = 8, i =0 \n" ;

   ostream& os = cout;

   pattern(os, 8, 0);
}

// Precondition: n is a power of 2 greater than zero.
// Postcondition: The longest line of the pattern has
//                n stars beginning in column i of the output.
// Base Case: The recursion stops if n == 0
void pattern(ostream& outs, unsigned int n, unsigned int i)
{
      if (n != 0)
      {
            pattern(outs, n / 2, i); //1st call starts printing ** at i

            for (int j = 0; j < i; j++)
                  outs << ' ';

            for (int k = 0; k < n; k++)
                  outs << "* ";

            cout << endl;

            pattern(outs, n / 2, i + n / 2); //2nd call to make ** go out further
      }
      else
            return;
}
```

Program Output - Screenshot

**Assignment4b.cpp running with n=8, i=0 **



**Assignment4b.cpp running with n=16, i=0 **