CS 1200 Homework 2 (Paper)

October 5th, 2020

Gian Zignago

---

1. John is the carpenter,
Bob is the painter,
And smith is the plumber

`Question_1.py` output:

```
Carpenter = John  Painter = Bob  Plumber = Smith
```

See `Question_1.py` on the next page

2. See `Question_2.cpp` on pages 3 & 4

Bonus: See `Question_2_Bonus.cpp` on pages 5 - 7

3. **Proof:** suppose a, b, c, and d such that

$$a|b \quad a|c \quad \text{and} \quad b|d$$

By the definition of divisibility, $b = ar, c = as$, and $d = bt$ for some integers *r*, *s*, and *t*. Then

Since $d = bt \quad and \quad b = ar, \quad then \quad d = (ar)t$ by substitution

$7c^2d - 3bc + 5d = 7(as)^2\big((ar)t\big) - 3(ar)(as) + 5((ar)t)$ by substitution

$= 7a^3s^2rt - 3a^2sr + 5art$ by distributive law of algebra

$= a^3(7s^2rt) - a^2(3sr) + a(5rt)$ by commutative law of algebra

$= a(a^2(7s^2rt) - a(3sr) + 5rt)$ by factoring

Let $m = (a^2(7s^2rt) - a(3sr) + 5rt)$. Then *m* is an integer, and thus $7c^2d - 3bc + 5d = am$ where *m* is an integer. By definition of divides, then, $a \mid (7c^2d - 3bc + 5d)$.

--Question_1.py--

```
people = ['Bob', 'John', 'Smith']

def distinct(L):
    if len(L) < 2:
        return True
    else:
        j = L.pop()
        if j in L:
            return False
        return distinct(L)

def earnsMore(p1,p2):
    global carpenter, painter, plumber

    if p1 == p2:
        return False
    if (p1 == 'John') and (p2 == 'Bob'): #Bob makes more than John
        return False
    if (p1 == painter) and (p2 == plumber): #plumber makes more than painter
        return False
    return True

def heardOf(p1, p2):
    global carpenter, painter, plumber
    if p2 == painter: #carpenter and plumber are each known by the other 2
        return False
    if(p1 == 'Smith') and (p2 == 'Bob'): #Smith has never heard of Bob
        return False
    return True

def solve2():
    global carpenter, painter, plumber
    for carpenter in people:
        for painter in people:
            for plumber in people:
                if distinct([carpenter, painter, plumber]):
                    sol = heardOf(painter, carpenter)
                    sol = sol and earnsMore('Smith', painter)
                    sol = sol and earnsMore(plumber, 'Bob')
                    sol = sol and heardOf(carpenter, plumber)
                    sol = sol and heardOf('Smith', carpenter)
                    if sol:
                        print("Carpenter =", carpenter,
                              " Painter =", painter, " Plumber =", plumber)
solve2()
```

--Question_2.cpp--

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

//the number to be tested for "magicness"
const unsigned long long int MAGIC_NUM = 93571393692802302;

int main()
{
    bool isMagic;
    unsigned long long int num = MAGIC_NUM;

    int i = 1; //records step number for output

    while(num != 1)
    {
        //outputs each step, starting at initial value
        cout << endl << "Step " << i << ": " << num;

        if(num % 2 == 0) { num /= 2; }

        else { num = num * 3 + 1; }

        i++;
    }

    //outputs final step (num should equal 1)
    cout << endl << "Step " << i << ": " << num;

    isMagic = ( num == 1 ? true : false );

    //results
    cout << endl << endl << "total steps elapsed: " << i
         << endl << MAGIC_NUM << (isMagic ? " is" : " is not")
         << " a magic number." << endl << endl;

    return 0;
}
```

--Question_2.cpp Output--

```
Step 696: 1084
Step 697: 542
Step 698: 271
Step 699: 814
Step 700: 407
Step 701: 1222
Step 702: 611
Step 703: 1834
Step 704: 917
Step 705: 2752
Step 706: 1376
Step 707: 688
Step 708: 344
Step 709: 172
Step 710: 86
Step 711: 43
Step 712: 130
Step 713: 65
Step 714: 196
Step 715: 98
Step 716: 49
Step 717: 148
Step 718: 74
Step 719: 37
Step 720: 112
Step 721: 56
Step 722: 28
Step 723: 14
Step 724: 7
Step 725: 22
Step 726: 11
Step 727: 34
Step 728: 17
Step 729: 52
Step 730: 26
Step 731: 13
Step 732: 40
Step 733: 20
Step 734: 10
Step 735: 5
Step 736: 16
Step 737: 8
Step 738: 4
Step 739: 2
Step 740: 1

total steps elapsed: 740
93571393692802302 is a magic number.
```

Note: The full output is too long to fit in this file

--Question_2_Bonus.cpp--

```cpp
#include <iostream>
using std::cin;
using std::cout;
using std::endl;

int main()
{
    unsigned long long upperLimit = 18446744073709000000; //maximum value for typ
e
    char limitChoice; //The user's choice on whether to enter a limit

    bool isMagic;
    bool counterFound = false;
    unsigned long long userNum = 1; //the number being tested, starts at 1
    unsigned long long num; //used for calculations on userNum

    //introductory message
    cout << endl << "Welcome to the Magic Number Counter Example searcher!"
         << endl << endl << "This program searches all numbers from 1 to a "
         << "given limit in search" << endl << "of numbers that serve as counter
examples "
         << "to the Collatz Conjecture." << endl << "Set a limit or let the progr
am run "
         << "until it finds a counter example!" << endl
         << endl << "Would you like to enter an upper limit? (y/n)" << endl;

    cin >> limitChoice;

    //if no, program runs until max value for type reached
    if((limitChoice == 'y') || (limitChoice == 'Y'))
    {
        cout << endl << "Enter a limit: ";
        cin >> upperLimit;
    }

    cout << endl << "Search in progress..." << endl
         << endl << "Numbers checked:" << endl;

    while((!counterFound) && (userNum <= upperLimit))
    {
        if(userNum >= 10000000) //outputs milestones (starting at 10 million)
        {
```

```
            if(userNum % 1000000000000 == 0)
                cout << userNum / 1000000000000 << " trillion" << endl;

            else if(userNum % 1000000000 == 0 )
                cout << userNum / 1000000000 << " billion" << endl;

            else if(userNum % 10000000 == 0)
                cout << userNum / 1000000 << " million" << endl;
        }

        int i = 1; //calculation step number

        num = userNum;

        //if calculation exceeds 3000 steps, number assumed to not be magic
        while((num != 1) && (i < 3000))
        {
            if(num % 2 == 0) { num /= 2; }

            else { num = num * 3 + 1; }

            i++;
        }

        isMagic = ( num == 1 ? true : false );

        if(!isMagic) //counter example
        {
            counterFound = true;
            cout << "Counter example found: " << userNum << endl;
        }

        userNum++;
    }

    cout << endl << "Upper Limit Reached" << endl << "Search ended" << endl << en
dl;

    return 0;
}
```

--Question_2_Bonus.cpp Output--

```
Welcome to the Magic Number Counter Example searcher!

This program searches all numbers from 1 to a given limit in search
of numbers that serve as counter examples to the Collatz Conjecture.
Set a limit or let the program run until it finds a counter example!

Would you like to enter an upper limit? (y/n)
y

Enter a limit: 50000000

Search in progress...

Numbers checked:
10 million
20 million
30 million
40 million
50 million

Upper Limit Reached
Search ended
```