Eli Sobylak
30-sept-15

Ch 1: 1.4, 1.5, 1.7, 1.8, 1.12

1.4
pgm1_4.cpp

```
/* This program will take a file a read through it to see if there is an #include
statement, it will then output the #include into a new file called output. I wasn't
quite sure what the question was asking or and I had to clarify with multiple
colleagues. I was not able to get the output file to have the whole file and the related
#include statements put in its place. */
#include <iostream>
#include <string>
#include <string.h>
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

using namespace std;

bool chkInclude(string in) {
	const char *cs = in.c_str();
	if (strncmp(cs, "#include", 8) == 0) {
		cout << "It was true" << endl;
		return true;
	}
	else {
		cout << "it was false" << endl;
		return false;
	}

}

int main(int argc, char** argv) {
	string s;
	ssize_t ssz;
	size_t sz;
	char *buff = NULL;

	if (argc != 3) {
		printf("Usage: openRW infile outfile\n");
		exit(1);
	}
	FILE *r_fd = fopen(argv[1], "r");
```

```
	FILE *w_fd = fopen(argv[2], "w");

	if (r_fd == NULL || w_fd == NULL) {
		printf("Error on open file(s)\n");
		exit(1);
	}

	while ((ssz = getline(&buff, &sz, r_fd)) > 0) {
		cout << "checking if true" << endl;
		if(chkInclude(buff) == true){
		fwrite(buff, ssz, sizeof(char), w_fd);
		cout << "Look in the " << argv[2] << " file for the results" << endl;
		}
	}

}
```

1.5
numOnes.cpp

```
/* This program will take any number N for an input and then output it in its binary
representation, when read from bottom to top. In addition, it will also output the
number of 1s in N */
#include <iostream>
using namespace std;


/*int userNum() {

	return n;
}
*/
int oneCounter = 1;

void numOnes(int n) {


	if(n == 0) {
		cout << "0\n";
	}
	else if(n == 1) {
		cout << "1\n";
		//oneCounter = oneCounter + 1;
		cout << "Number of ones in the binary representation of N is: " <<
oneCounter << "\n";
	}
	else {
```

```
		cout << n % 2 << "\n";
		if(n % 2 == 1) {
			oneCounter = oneCounter + 1;
		}
		//oneCounter = oneCounter + 1;
		numOnes(n/2);
		//cout << n % 2 << "\n";
		//numOnes(n%2);
		//cout << main(n%2);
	}
}

int main() {
	int n;
	cout << "Enter a number: ";
	cin >> n;
	//int n = 5;
	numOnes(n);
	return 0;
}
```

See hw2_part2_sobylak.txt for problems 1.7,1.8,1.12

OS Homework 1

1.8

a. $\sum_{i=0}^{\infty} \frac{1}{4^i}$

$$\sum_{i=0}^{\infty} \frac{1}{4^i} = \frac{1}{1} + \frac{1}{4} + \frac{1}{16} + \frac{1}{64} \quad \frac{1}{256}$$

(1)

$(4^0) \quad (4^1) \quad (4^2) \quad (4^3) \quad (4^4)$

3 terms:

$$1 + \frac{1}{4} = \frac{5}{4}$$

$$1 + \frac{1}{4} + \frac{1}{16} = \frac{21}{16}$$

$$1 + \frac{1}{4} + \frac{1}{16} + \frac{1}{64} = \frac{85}{64}$$

$$1 + \frac{1}{4} + \frac{1}{16} + \frac{1}{64} + \frac{1}{256} = \frac{341}{256}$$

partial sums:

$$\left(1, \frac{5}{4}, \frac{21}{16}, \frac{85}{64}, \frac{341}{256}, \ldots\right)$$

so,

$$\sum_{i=0}^{\infty} \frac{1}{4^i} = 1.\overline{33} \qquad \sum_{n=0}^{\infty} \frac{1}{4^n}$$

6. $\sum_{i=0}^{\infty} \frac{i}{4^i}$

Sums: $$\sum_{i=0}^{\infty} \frac{i}{4^i} = \underset{(4^0)}{\overset{(0)}{\frac{0}{1}}} + \underset{(4^1)}{\frac{1}{4}} + \underset{(4^2)}{\overset{\left(\frac{1}{8}\right)}{\frac{2}{16}}} + \underset{(4^3)}{\frac{3}{64}} + \underset{(4^4)}{\overset{\left(\frac{1}{64}\right)}{\frac{4}{256}}}$$

$$0 + \frac{1}{4} = \frac{1}{4}$$

$$0 + \frac{1}{4} + \frac{1}{8} = \frac{3}{8}$$

$$0 + \frac{1}{4} + \frac{1}{8} + \frac{3}{64} = \frac{27}{64}$$

$$0 + \frac{1}{4} + \frac{1}{8} + \frac{3}{64} + \frac{4}{64} = \frac{7}{16}$$

Partial Sums:

$$\left(\frac{1}{4}, \frac{3}{8}, \frac{27}{64}, \frac{7}{16}, \ldots\right)$$

$$\sum_{i=0}^{\infty} \frac{i}{4^i} = .\overline{4}$$

1.12 Prove the following

a) $\sum_{i=1}^{n}(2i-1) = n^2$

let $P(n)$ be $\sum_{i=1}^{n}(2i-1) = n^2$

1. Base case; $n = 1$

Prove: $\sum_{i=1}^{1}(2i-1) = 1^2$

Proof: $\sum_{i=1}^{1}(2(1)-1) = 1$ and $1^2 = 1$

thus, $P(1)$.

2. Induction Step; let $n \in \mathbb{N}$, we fix arbitrary integer $n \geq 1$ to show that $P(n+1)$ is true

i) Assume $P(n)$: $\sum_{i=1}^{n}(2i-1) = n^2$ for some int $n \geq 1$

ii) Induction:

$$\sum_{i=1}^{n+1}(2i-1) = (n+1)(n^2)$$

Proof: Sum the left hand side

$$\sum_{i=1}^{n+1}(2i-1) = 0+1+2+3+\ldots+n+(n+1)$$
$$= (0+1+2+3+\ldots+n)+(n+1)$$
$$= \left(\sum_{i=1}^{n}(2i-1)\right)+(n+1)$$

$= [n^2] + (n+1)$.

$= n^2 + n + 1$

$= \cancel{n(n+1)+1}$

$= n^2 + n + 1$

by induction it follows

$$\sum_{i=1}^{n} (2i-1) = n^2$$

b. Prove $\sum_{i=1}^{n} i^3 = \left(\sum_{i=1}^{n} i\right)^2$

~~proof~~ let $P(n)$ be $\sum_{i=1}^{n} i^3 = \left(\sum_{i=1}^{n} i\right)^2$

1. Base case; $n=1$

Prove $\sum_{i=1}^{1} i^3 = \left(\sum_{i=1}^{1} i\right)^2$

Proof $\sum_{i=1}^{1} 1^3 = 1$ and $\left(\sum_{i=1}^{1} 1\right)^2 = 1$

thus, $P(1)$.

next page

2. Induction Step; let $n \in \mathbb{N}$ and we ~~fix~~ ~~an~~ arbitrary integer $n \geq 1$. Then we'll show that $P(n+1)$ is also true.

i) Assume $P(n)$: $\sum_{i=1}^{n} i^3 = \left(\sum_{i=1}^{n} i\right)^2$ for some int $n \geq 1$

ii) use induction:

$$\sum_{i=1}^{n+1} i^3 = (n+1)\left(\sum_{i=1}^{n} i\right)^2$$

proof; write sum on the lhs ~~to~~ be proven

$$\sum_{i=1}^{n+1} i^3 = 1 + 4 + 9 + 16 + \ldots + n + (n+1)$$
$$= (1 + 4 + 9 + 16 + \ldots + n) + (n+1)$$
$$= 1 + 8 + 27 + 64 + \ldots + n + (n+1)$$
$$= (1 + 8 + 27 + 64 + \ldots + n) + (n+1)$$
$$= \left(\sum_{i=1}^{n} i^3\right) + (n+1)$$
$$= \left[\left(\sum_{i=1}^{n} i\right)^2\right] + (n+1)$$

by induction it follows

$$\sum_{i=1}^{n} i^3 = \left(\sum_{i=1}^{n} i\right)^2$$

1.7 Prove the following

a. $\log X < X$ for all $X > 0$

It is true that $0 < x \leq 1$

if $x < 1$ then $\log x < 0$

and

if $x = 1$ then $\log x = 0$

now we use induction to show $x > 1$ is true

Base case;

if $1 < x < 2$ then $\log x < 1$

and

if $x = 2$ then $\log x = 1$

$\therefore 1 < x \leq 2$

Inductive Hypothesis;

Assume true $P < x \leq 2P$

prove $2P < y \leq 4P$ & this is because y is between the limits

$2(P) < y \leq 2(2P)$

or

$P < \frac{y}{2} \leq 2P$

$\log y = \log 2 + \log \frac{y}{2}$

$= \log y = 1 + \log \frac{y}{2}$

this follows that

$\log \frac{y}{2} < \frac{y}{2} \implies \log y < 1 + \frac{y}{2}$

substitute

$\frac{y}{2}$ will always be greater than 1

$\therefore \log y < \frac{y}{2} + \frac{y}{2} = \log y \leq y$

6. $\log_c(A^B) = B\log_c A$

1. let $\log_c(A^B) = x$ Reason: $\log(A^B)$ must be a number

2. $C^x = A^B$ Reason: exponentiated form of #1

3. $(C^x)^{1/B} = (A^B)^{1/B}$ Reason: exponents if $A=B$ then $A^N = B^N$

4. $C^{x/B} = A$ Reason: exponents $(A^P)^N = A^{P \cdot N}$

5. $\log_c(A) = \frac{x}{B}$ Reason: equivalent log form of #4

6. $B \cdot \log_c(A) = x$ Reason: if $A=B$ then $R \cdot A = R \cdot B$

7. $B \cdot \log_c(A) = \log_c(A^B)$ Reason: substitute from #1 for x in #6

Here are my solutions for the simple c++ problems assigned to us. I hope to come back to them once I have a better grasp on the c++ language.

IntCell.h

```
#ifndef IntCell_H
#define IntCell_H

/**
 * A class for simulating an integer memory cell.
 */
class IntCell
{
  public:
    explicit IntCell( int initialValue = 0 );
    int read( ) const;
    void write( int x );
    int add(IntCell* x);
    int sub(IntCell* x);
    int addAndReturnIntCell(int y);

  private:
    int storedValue;
};

#endif
```

Intcell.cpp

```
#include "IntCell.h"

/* So, this is my generalized summary of what I was able to accomplish for this
exercise.
As you'll notice my code will not be runable in this state and this would be due to
the fact
that while doing hw1_question1.5 I accidentally dumped a bunch of hex into my
original Intcell.cpp
file and overwrote it without realizing. Here's what I was able to reproduce. */

/**
 * Construct the IntCell with initialValue
 */
IntCell::IntCell( int initialValue ) : storedValue( initialValue )
{
}

/**
 * Return the stored value.
 */
int IntCell::read( ) const
{
   return storedValue;
}

/**
 * Store x.
 */
void IntCell::write( int x )
{
   storedValue = x;
}

int Intcell::add (/*parameter was a pointer to an Intcell*/) {
	/*Here I had the add function read the stored value from the Intcell
	calling the function and add it to the read value of the incoming
	Intcell*/
}

int Intcell::sub (/*paramater was a pointer to an Intcell*/) {
	/*This was the same code as th addition function but had the minus operator
instead of
	addition*/
}
```

```
int Intcell::addAndReturnIntcell(/*paramater here was an int, say y*/) {
	/*Since I passed it a value for the int y, I was able to add it to an existing
Intcell
	and although I never got it to be able to return an Intcell, I was able to output
	the result of the constant y and the Intcell*/
}
```

TestIntCell.cpp

```
#include <iostream>
#include "IntCell.h"
using namespace std;

int main( )
{
   IntCell m;   // Or, IntCell m( 0 ); but not IntCell m( );
   IntCell n;

   int y = 12;

   m.write( 5 );
   n.write( 7 ) ;
   cout << "Cell M contains: " << m.read( ) << endl;
   cout << "Cell N contains: " << n.read( ) << endl;
   cout << "N + M added: " << n.add(&m) << endl;
   cout << "N - M subtracted: " << n.sub(&m) << endl;
   cout << "N + int y, which is 12: " << n.addAndReturnIntCell(y) << endl;
   return 0;
}
```