Homework 8

Source Code

Main.cpp

// Preston Knibbe

// 11/20/15

#include "Binary.h"

#include <iostream>

#include <math.h>

using namespace std;

int main()

{

Binary number1;

Binary number2;

cout << "Enter a binary number: " << endl;

cin >> number1;

cout << "Enter another binary number: " << endl;

cin >> number2;

Binary sum;

Binary difference;

Binary multTotal;

Binary divTotal;

sum = number1 + number2;

cout << "Sum: " << sum << endl;

difference = number1 - number2;

cout << "Difference: " << difference << endl;

multTotal = number1 \* number2;

cout << "Multiplied: " << multTotal << endl;

divTotal = number1 / number2;

cout << "Divided: " << divTotal << endl;

/\*cout << "Sum: " << sum.binaryToDecimal(sum.getBinary()) << endl;

cout << "Difference: " << difference.binaryToDecimal(difference.getBinary()) << endl;

cout << "Multiplication Total: " << multTotal.binaryToDecimal(multTotal.getBinary()) << endl;

cout << "Division Total: " << divTotal.binaryToDecimal(divTotal.getBinary()) << endl;\*/

return 0;

}

Binary.h

// Preston Knibbe

// 11/20/15

#ifndef BINARY\_H

#define BINARY\_H

#include <iostream>

using namespace std;

class Binary

{

public:

Binary();

Binary(string);

string getBinary();

void setBinary(string bin);

int binaryToDecimal(string);

string decimalToBinary(int);

Binary operator+(Binary binary1);

Binary operator-(Binary binary1);

Binary operator/(Binary);

Binary operator\*(Binary);

private:

string binaryNum;

friend istream& operator>>(istream&, Binary&);

friend ostream& operator<<(ostream&, Binary&);

};

inline istream& operator>>(istream& input, Binary& info) {

input >> info.binaryNum;

return input;

}

inline ostream& operator<<(ostream& output, Binary& info) {

output << info.binaryNum;

return output;

}

#endif // BINARY\_H

Binary.cpp

// Preston Knibbe

// 11/20/15

#include "Binary.h"

#include <iostream>

#include <math.h>

using namespace std;

Binary::Binary() {

}

Binary::Binary(string num)

{

binaryNum = num;

}

string Binary::getBinary()

{

return binaryNum;

}

void Binary::setBinary(string bin)

{

binaryNum = bin;

}

int Binary::binaryToDecimal(string bin)

{

int multiplier = 0;

int total = 0;

int length = bin.length() -1;

for(int i = length; i >= 0; i--) {

if (bin[i] == '1') {

total += pow(2,multiplier);

multiplier++;

} else {

multiplier++;

}

}

return total;

}

string Binary::decimalToBinary(int dec1)

{

string newBinary = "";

bool startTest = true;

double dec = dec1;

double num = 4096.0;

double startNum;

while (startTest) {

if (dec >= num && dec <= 511) {

startNum = num;

startTest = false;

} else {

num /= 2;

}

}

while (dec > 0) {

while (startNum >= 1) {

if ((dec / startNum) >= 1) {

newBinary += "1";

dec -= startNum;

startNum /= 2;

} else {

newBinary += "0";

startNum /= 2;

}

}}

return newBinary;

}

Binary Binary::operator+(Binary binary1)

{

Binary newObject;

int a = binaryToDecimal(binaryNum);

int b = binaryToDecimal(binary1.binaryNum);

int c = a + b;

newObject.binaryNum = decimalToBinary(c);

return(newObject);

}

Binary Binary::operator-(Binary binary1)

{

Binary newObject;

int a = binaryToDecimal(binaryNum);

int b = binaryToDecimal(binary1.binaryNum);

int c = a - b;

newObject.binaryNum = decimalToBinary(c);

return(newObject);

}

Binary Binary::operator\*(Binary binary1)

{

Binary newObject;

double a = binaryToDecimal(binaryNum);

double b = binaryToDecimal(binary1.binaryNum);

double c = a \* b;

newObject.binaryNum = decimalToBinary(c);

return(newObject);

}

Binary Binary::operator/(Binary binary1)

{

Binary newObject;

double a = binaryToDecimal(binaryNum);

double b = binaryToDecimal(binary1.binaryNum);

double c = a / b;

newObject.binaryNum = decimalToBinary(c);

return(newObject);

}

Code Output

1)

Enter a binary number:

10011

Enter another binary number:

1100

Sum: 11111

Difference: 111

Multiplied: 11100100

Divided: 1

Process returned 0 (0x0) execution time : 9.329 s

Press any key to continue.

2)

Enter a binary number:

1111

Enter another binary number:

101

Sum: 10100

Difference: 1010

Multiplied: 1001011

Divided: 11

Process returned 0 (0x0) execution time : 7.708 s

Press any key to continue.

3)

Enter a binary number:

10010

Enter another binary number:

1001

Sum: 11011

Difference: 1001

Multiplied: 10100010

Divided: 10

Process returned 0 (0x0) execution time : 9.696 s

Press any key to continue.