Object Oriented Systems Design

Assignment for Theory [MA31011] & Lab [MA39011]

Course instructor: Prof. Bodhayan Roy

Name: Maitreyi Swaroop

Roll No. : 19MA20065

Date of submission. : November 18, 2021

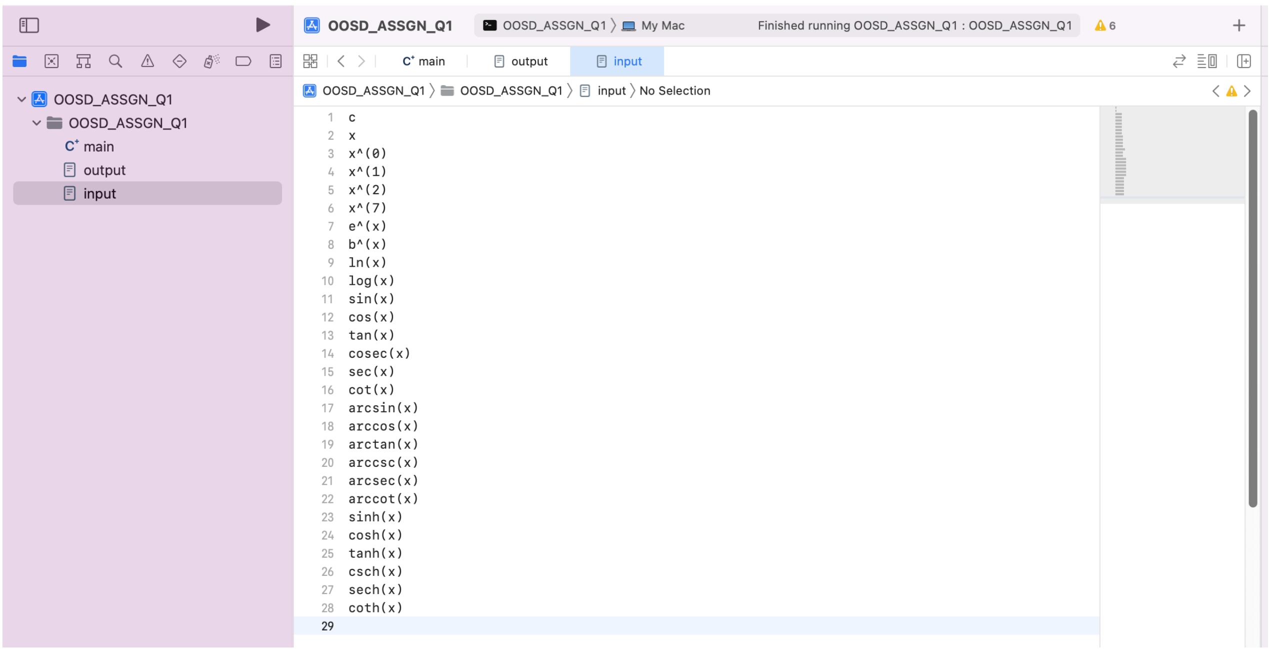
Contents

1 Question statements, input and output

Question 1

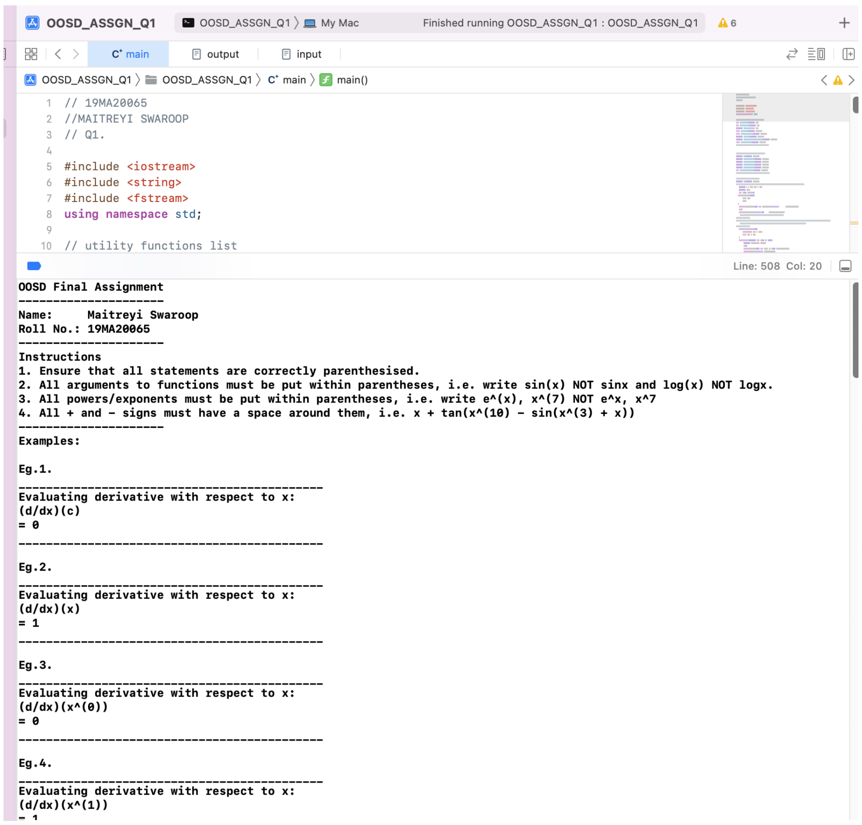
Question statement. Differentiate a given function with respect to x. The input will consist of only one line, given in the file input.txt. The input will be just a power, exponential, logarithmic, hyperbolic, trigonometric or an inverse trigonometric function of x. The output should be displayed in the terminal. You may find a list of such functions here: `http://www.math.com/tables/derivatives/tableof.htm`.

Screenshots of input file and corresponding output Although the assignment instructions mention the input containing only one line, for the purpose of demonstrating the code for all functions, here I have attached screenshots of the code taking multiple line inputs, with each line being an expression to be differentiated. (The .cpp file submitted takes in only one line of input).


OOSD_ASSGN_Q1
OOSD_ASSGN_Q1 〉 My Mac
Finished running OOSD_ASSGN_Q1 : OOSD_ASSGN_Q1
6
main
output
input
OOSD_ASSGN_Q1 〉 OOSD_ASSGN_Q1 〉 input 〉 No Selection
OOSD_ASSGN_Q1
OOSD_ASSGN_Q1
main
output
input

```
c
x
x^(0)
x^(1)
x^(2)
x^(7)
e^(x)
b^(x)
ln(x)
log(x)
sin(x)
cos(x)
tan(x)
cosec(x)
sec(x)
cot(x)
arcsin(x)
arccos(x)
arctan(x)
arccsc(x)
arcsec(x)
arccot(x)
sinh(x)
cosh(x)
tanh(x)
csch(x)
sech(x)
coth(x)
```

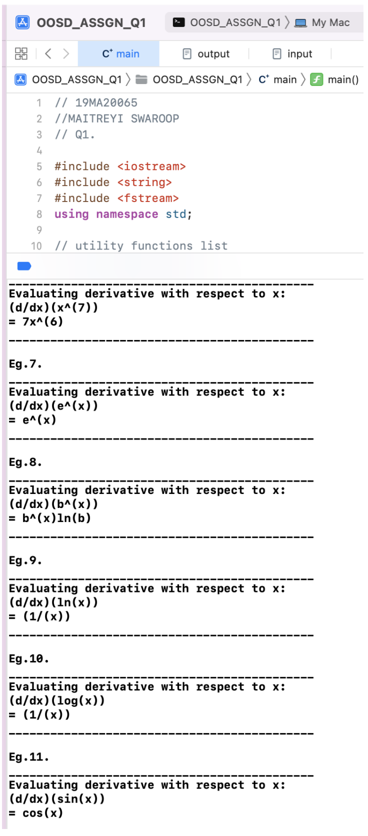
Figure 1: Q1. Input file (input.txt)



```
// 19MA20065
//MAITREYI SWAROOP
// Q1.

#include <iostream>
#include <string>
#include <fstream>
using namespace std;

// utility functions list
```

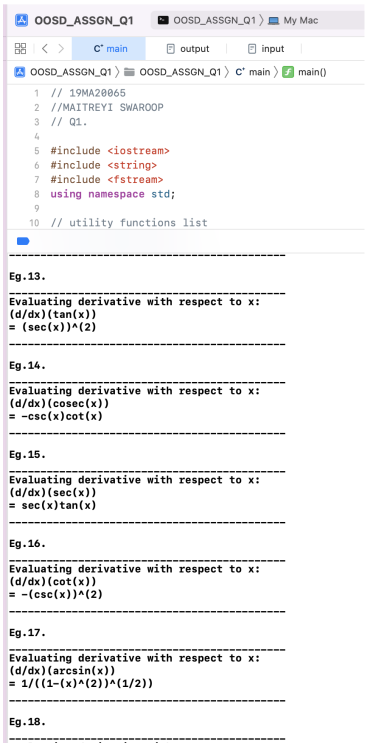
```
OOSD Final Assignment
---------------------
Name:     Maitreyi Swaroop
Roll No.: 19MA20065
---------------------
Instructions
1. Ensure that all statements are correctly parenthesised.
2. All arguments to functions must be put within parentheses, i.e. write sin(x) NOT sinx and log(x) NOT logx.
3. All powers/exponents must be put within parentheses, i.e. write e^(x), x^(7) NOT e^x, x^7
4. All + and - signs must have a space around them, i.e. x + tan(x^(10) - sin(x^(3) + x))
---------------------
Examples:

Eg.1.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(c)
= 0

--------------------------------------------

Eg.2.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(x)
= 1

--------------------------------------------

Eg.3.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(x^(0))
= 0

--------------------------------------------

Eg.4.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(x^(1))
= 1
```

Figure 2: Q1. Output



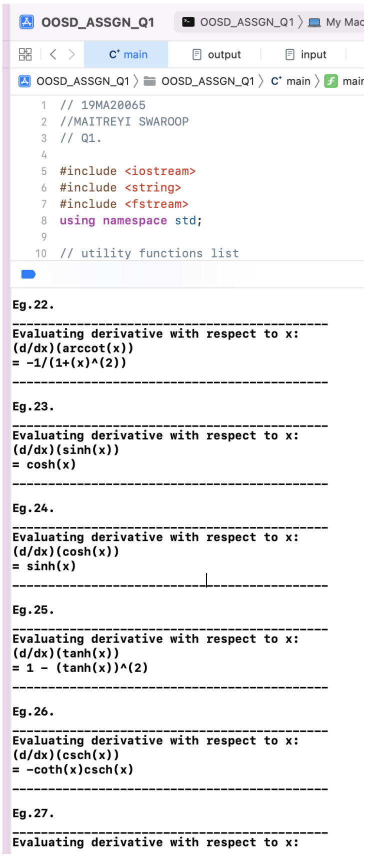
```
// 19MA20065
//MAITREYI SWAROOP
// Q1.

#include <iostream>
#include <string>
#include <fstream>
using namespace std;

// utility functions list
```

```
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(x^(7))
= 7x^(6)

--------------------------------------------

Eg.7.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(e^(x))
= e^(x)

--------------------------------------------

Eg.8.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(b^(x))
= b^(x)ln(b)

--------------------------------------------

Eg.9.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(ln(x))
= (1/(x))

--------------------------------------------

Eg.10.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(log(x))
= (1/(x))

--------------------------------------------

Eg.11.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(sin(x))
= cos(x)
```



```
// 19MA20065
//MAITREYI SWAROOP
// Q1.

#include <iostream>
#include <string>
#include <fstream>
using namespace std;

// utility functions list
```

```
--------------------------------------------

Eg.13.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(tan(x))
= (sec(x))^(2)

--------------------------------------------

Eg.14.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(cosec(x))
= -csc(x)cot(x)

--------------------------------------------

Eg.15.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(sec(x))
= sec(x)tan(x)

--------------------------------------------

Eg.16.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(cot(x))
= -(csc(x))^(2)

--------------------------------------------

Eg.17.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(arcsin(x))
= 1/((1-(x)^(2))^(1/2))

--------------------------------------------

Eg.18.
--------------------------------------------
```



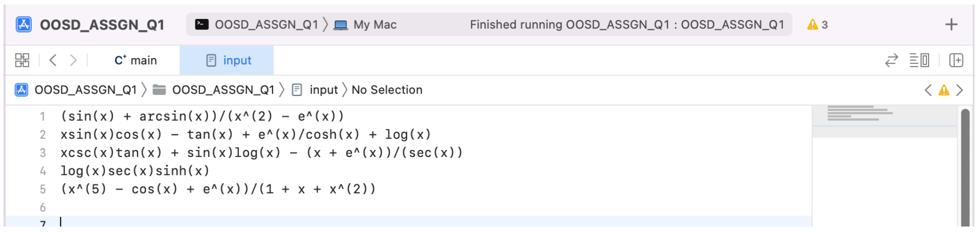
```
// 19MA20065
//MAITREYI SWAROOP
// Q1.

#include <iostream>
#include <string>
#include <fstream>
using namespace std;

// utility functions list
```

```
Eg.22.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(arccot(x))
= -1/(1+(x)^(2))

--------------------------------------------

Eg.23.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(sinh(x))
= cosh(x)

--------------------------------------------

Eg.24.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(cosh(x))
= sinh(x)

--------------------------------------------

Eg.25.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(tanh(x))
= 1 - (tanh(x))^(2)

--------------------------------------------

Eg.26.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(csch(x))
= -coth(x)csch(x)

--------------------------------------------

Eg.27.
--------------------------------------------
Evaluating derivative with respect to x:
```

Question 2

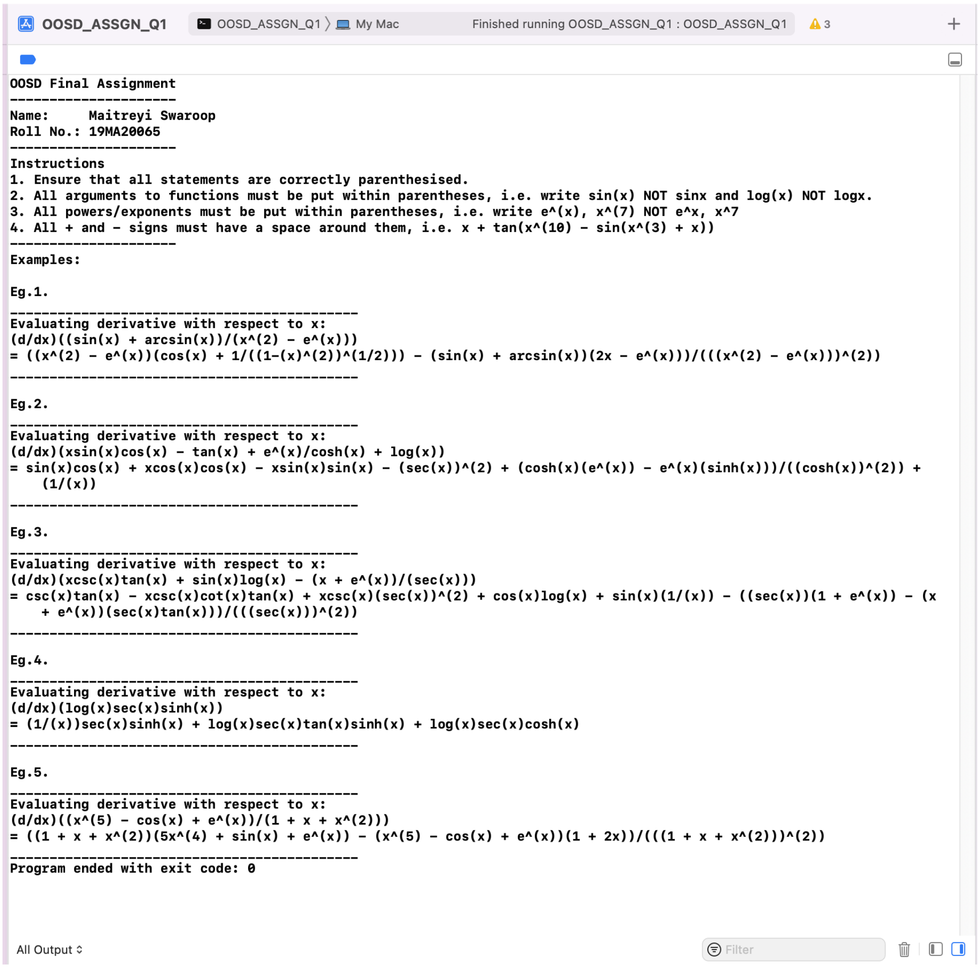
Question statement. Differentiate with respect to x any expression generated by sums, differences, products and quotients of the above functions. For example, the given expression may be of the form $(sin(x) + sin^{-1}(x))/(x^2 - e^x)$.

Screenshots of input file and corresponding output Five input statements entered. The submitted file only takes one line as input.



```
(sin(x) + arcsin(x))/(x^(2) - e^(x))
xsin(x)cos(x) - tan(x) + e^(x)/cosh(x) + log(x)
xcsc(x)tan(x) + sin(x)log(x) - (x + e^(x))/(sec(x))
log(x)sec(x)sinh(x)
(x^(5) - cos(x) + e^(x))/(1 + x + x^(2))


```

Figure 3: Q2. Input file (input.txt)



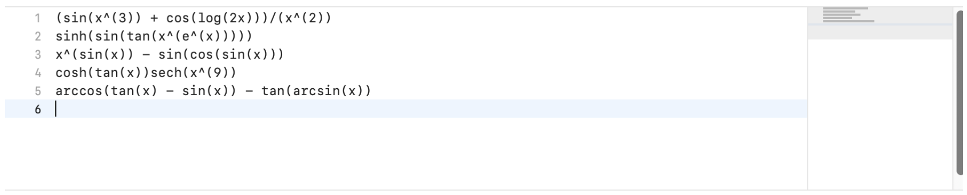
```
OOSD Final Assignment
---------------------
Name:     Maitreyi Swaroop
Roll No.: 19MA20065
---------------------
Instructions
1. Ensure that all statements are correctly parenthesised.
2. All arguments to functions must be put within parentheses, i.e. write sin(x) NOT sinx and log(x) NOT logx.
3. All powers/exponents must be put within parentheses, i.e. write e^(x), x^(7) NOT e^x, x^7
4. All + and - signs must have a space around them, i.e. x + tan(x^(10) - sin(x^(3) + x))
---------------------
Examples:

Eg.1.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)((sin(x) + arcsin(x))/(x^(2) - e^(x)))
= ((x^(2) - e^(x))(cos(x) + 1/((1-(x)^(2))^(1/2))) - (sin(x) + arcsin(x))(2x - e^(x)))/(((x^(2) - e^(x)))^(2))
--------------------------------------------

Eg.2.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(xsin(x)cos(x) - tan(x) + e^(x)/cosh(x) + log(x))
= sin(x)cos(x) + xcos(x)cos(x) - xsin(x)sin(x) - (sec(x))^(2) + (cosh(x)(e^(x)) - e^(x)(sinh(x)))/((cosh(x))^(2)) +
    (1/(x))
--------------------------------------------

Eg.3.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(xcsc(x)tan(x) + sin(x)log(x) - (x + e^(x))/(sec(x)))
= csc(x)tan(x) - xcsc(x)cot(x)tan(x) + xcsc(x)(sec(x))^(2) + cos(x)log(x) + sin(x)(1/(x)) - ((sec(x))(1 + e^(x)) - (x
    + e^(x))(sec(x)tan(x)))/(((sec(x)))^(2))
--------------------------------------------

Eg.4.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(log(x)sec(x)sinh(x))
= (1/(x))sec(x)sinh(x) + log(x)sec(x)tan(x)sinh(x) + log(x)sec(x)cosh(x)
--------------------------------------------

Eg.5.
--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)((x^(5) - cos(x) + e^(x))/(1 + x + x^(2)))
= ((1 + x + x^(2))(5x^(4) + sin(x) + e^(x)) - (x^(5) - cos(x) + e^(x))(1 + 2x))/(((1 + x + x^(2)))^(2))
--------------------------------------------
Program ended with exit code: 0
```

Figure 4: Q2. Output

Question 3

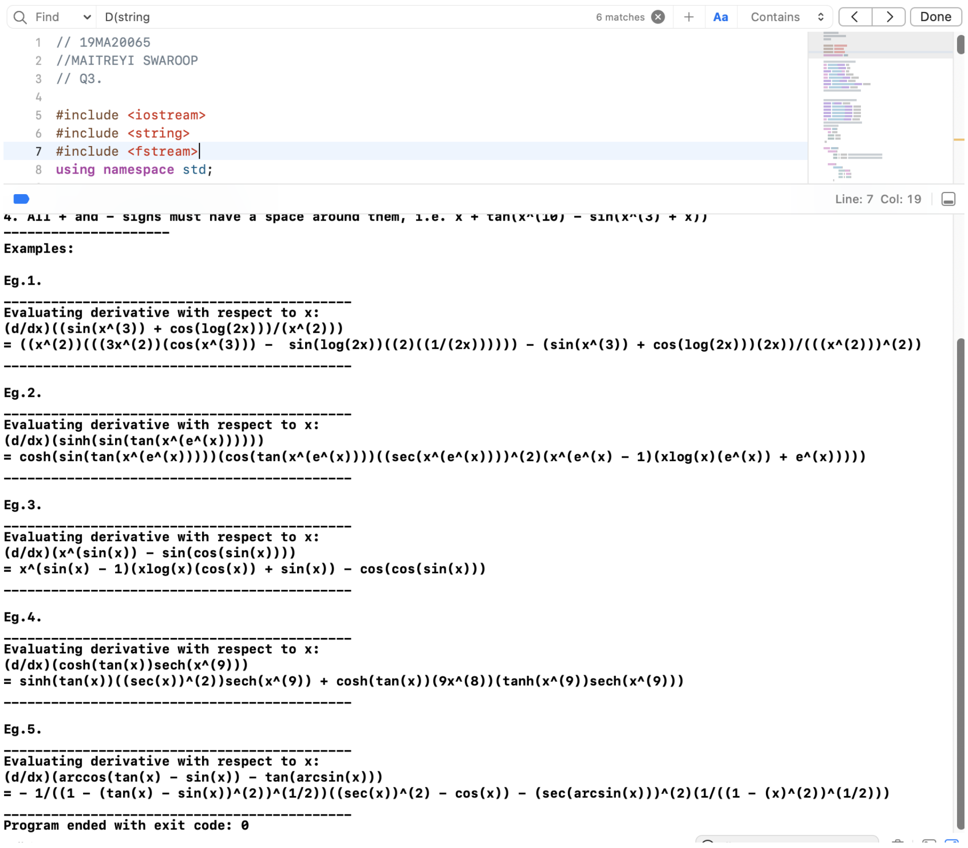
Question statement. Differentiate with respect to x any expression formed by the composition of the above functions implementing the chain rule of derivatives. For example, an expression may be of the form: $(sin(x^3) + cos(log(2x)))/(x^2)$.

Screenshots of input file and corresponding output Five input statements entered. The submitted file only takes one line as input.



```
(sin(x^(3)) + cos(log(2x)))/(x^(2))
sinh(sin(tan(x^(e^(x)))))
x^(sin(x)) - sin(cos(sin(x)))
cosh(tan(x))sech(x^(9))
arccos(tan(x) - sin(x)) - tan(arcsin(x))

```

Figure 5: Q3. Input file (input.txt)



```
// 19MA20065
//MAITREYI SWAROOP
// Q3.

#include <iostream>
#include <string>
#include <fstream>
using namespace std;
```

```
4. All + and - signs must have a space around them, i.e. x + tan(x^(10) - sin(x^(3) + x))
---------------------
Examples:

Eg.1.

--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)((sin(x^(3)) + cos(log(2x)))/(x^(2)))
= ((x^(2))(((3x^(2))(cos(x^(3))) -  sin(log(2x))((2)((1/(2x)))))) - (sin(x^(3)) + cos(log(2x)))(2x))/(((x^(2)))^(2))
--------------------------------------------

Eg.2.

--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(sinh(sin(tan(x^(e^(x))))))
= cosh(sin(tan(x^(e^(x)))))(cos(tan(x^(e^(x))))((sec(x^(e^(x))))^(2)(x^(e^(x) - 1)(xlog(x)(e^(x)) + e^(x)))))
--------------------------------------------

Eg.3.

--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(x^(sin(x)) - sin(cos(sin(x))))
= x^(sin(x) - 1)(xlog(x)(cos(x)) + sin(x)) - cos(cos(sin(x)))
--------------------------------------------

Eg.4.

--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(cosh(tan(x))sech(x^(9)))
= sinh(tan(x))((sec(x))^(2))sech(x^(9)) + cosh(tan(x))(9x^(8))(tanh(x^(9))sech(x^(9)))
--------------------------------------------

Eg.5.

--------------------------------------------
Evaluating derivative with respect to x:
(d/dx)(arccos(tan(x) - sin(x)) - tan(arcsin(x)))
= - 1/((1 - (tan(x) - sin(x))^(2))^(1/2))((sec(x))^(2) - cos(x)) - (sec(arcsin(x)))^(2)(1/((1 - (x)^(2))^(1/2)))
--------------------------------------------
Program ended with exit code: 0
```

Figure 6: Q3. Output

2 Approach (brief explanation)

Every expression (`expr`) is viewed recursively as made up of subsequent expressions/*sub*expressions. The key functions in the code are:

1. `D(string expr)` This function performs the most basic differentiation on individual functions. The input expression is split into function `f` and argument `arg`. Following the differentiation rule of the given function, it returns the basic output.

2. `derivMain(string expr)` This function parses the expression and breaks it up into its constituent subexpressions, recursively calling upon itself (and other calculus functions such as derivQuot etc.) till it reaches an individual function and calls upon `D`. If the argument (`arg`) is itself an expression (i.e. the function is a composite function), it appends to the output of `D` the output of `derivMain(arg)`.

3. `derivQuot(string expr)` This function is called upon by `derivMain` when it encounters a `/`. It breaks the expression into the `numerator` and `denominator`, calls upon `derivMain` for each, then combines them in accordance with the quotient rule.

4. `derivProd(string expr)` This function is called upon by `derivMain` when it encounters a product of expressions. It breaks the expression into all its constituent subexpressions, calls upon `derivMain` for each, then combines them in accordance with the product rule.

5. `derivExpn(string expr)` This function is called upon by `derivMain` when it encounters an expression raised to another expression or for terms of the form $f(x)^{g(x)}$. It breaks the expression into all its constituent subexpressions, calls upon `derivMain` for each, then combines them in accordance with the exponent rule.

3 Function list and required input format

Here is the list of functions and how they should be referred to in the input string.

Function	**Refer to as** (in input string)	**Avoid**
constant	any character except `x`	
x	any character, eg. `a, b , c,...`	
x^n	`x^(n)`	non-integer powers, `x^n`
Exponential e^x or b^x	`e^(x)`, `b^(x)`	`e^x`, `b^x`
Logarithmic (only base e)	`log(x), ln(x)`	`logx, lnx`
Trigonometric	`sin(x), cos(x), tan(x)`	`sinx, cosx, tanx`
	`cosec(x)/csc(x), sec(x), cot(x)`	`cscx, secx, cotx`
Inverse Trigonometric	`arcsin(x), arccos(x), arctan(x),...`	`sin-1(x), cos-1(x), tan-1(x)`
Hyperbolic	`sinh(x), cosh(x), tanh(x)`	`sinhx, coshx, tanhx`
	`cosech(x)/csch(x), sech(x), coth(x)`	`cschx, sechx, cothx`

Table 1: Admissable input format.

Instructions

1. Ensure that all statements are correctly parenthesised.
2. All arguments to functions must be put within parentheses, i.e. write `sin(x)` NOT `sinx` and `log(x)` NOT `logx`.
3. All powers/exponents must be put within parentheses, i.e. write `e^(x)`, `x^(7)`, NOT `e^x`, `x^(7)`. Also, expressions like $sin^2(x)$ (== $sin(x)$ whole squared) must be entered as `(sin(x))^(2)`. Same for cos/tan/log etc.
4. All + and - signs must have a space around them, i.e. x + tan(x^(10) - sin(x^(3) + x)) (Do not worry too much about this point however, there is a utility function to correct lack of padding around + and - signs.)
5. Be careful about stray spaces too (eg. avoid `e ^(x)`), though there are checks for some cases.