**Simple Interpreter Project**

**Team Members:**

1- Ahmed IhabElselmy 6449  
2-Nada Samy 6357  
3-Mariam Mohamed Elraggal 6429

**1-Project Description:**

This is an implementation of a simple interpreterthat executes instructions and prints the result of execution. The program accepts a text file (.txt), reads the data inside and prints the data that is going to be processed. The file contains lines of equations. After the program checks each line, if an equation is not valid the equation is printed and is stated that it is invalid then the project terminates.

If all equations are valid then the program calculates the value of each variable and prints variables twice:

1- First time after sorting them with their variable name.

2- Second time after sorting them with the variable values.

**2- Function Description:**

***Stack \* initialize();***It initialize stack so that there are no elements inserted.

***intisEmpty(Stack \*s);***It check if stack is empty (return 1)or not (return -1) .

***Item top(Stack \*s);***  
It returns top element of stack without affecting it.

**Item pop(Stack \*s);**  
Removes an item from the stack and return it. The items are popped in the reversed order in which they are pushed. If the stack is empty, then it is said to be an Underflow condition.

**void push(Stack \*s, Item val);**Adds an item in the stack.

**intprecidence(char b);**Returns the precedence of the operator.

**voidinfixToPostfix(char\* infix, char\* postfix);**  
Converts the input infix expression to postfix expression.

**floatevaluatePostfix(char\* postfix);**  
Takes the postfix expression as an input and evaluate the result.

**voidreplaceNewLineBySpace(char \*s);**

It searches the line given if it contains a new line and if found it replaces it by a space.

**intRemoveSpaces(char\* token,int\* len,char\* temp);**  
Removes all spaces from the expression that is taken from file.

**intcheckEquation(char \*equation);**  
Checks the validity of taken expression from file.It returns 0 if expression is invalid.

**voidinsertCharacter(char \* a,charch,int p);**Inserts a space in the target place .

**intisnumber(char \*token);**Checks if token is number or not. It returns 1 if token is a number and 0 otherwise.

**struct Node \* newNode(char data[ ],float result);**  
Creates new node to binary search tree.

**struct Node\* put(struct Node\* node,float result, char data[ ]);**Insertsa new node with given result and data in binary search tree.

**voidInOrdertraverse(struct Node\* t);**

Sorts the binary tree by ascending order starting form left most node to root then to right most node. As nodes in the left sub-tree of the root has greater value than root and right sub-tree of the root has lesser value than root.

**char \*replaceWord(char \*s,char \*symbol,char \*value);**

Takes each line in file one by one and changes every unknown variable so there is only one unknown which will be by the left hand side of the ‘=’ sign. This prepares the equation to be ready for execution.

*Example:*

*File Contents:*

*X=12*

*Y=X+2*

*Then, in line 2 the equation with X and Y, Y will remain as it is ( an unknown ) while X will be replaced by its value then the function will return the new equation Y=12+2.*

**voidPrintData(Data lines[ ],int counter);**Prints all data the symbol and its value.

**void create(float heap[ ]);**

Creates the heap using the passed array parameter and calls upon the function**void down\_adjust(float heap[ ],inti);**

to sort the heap while it creates it.

**voiddown\_adjust(float heap[ ],inti);**

Sorts the heap by sorting 2 elements at a time.

**void Load();**  
The main function that we call in the main   
it loads data from file and check if file exist or not and check every line scanned if it is valid expression or not by function **intcheckEquation(char \*equation);** if the expression is invalid the program will exit and show an error message   
if the expression is valid we remove all spaces in by function.

**intRemoveSpaces(char\* token,int\* len,char\* temp);**

Also, as it reads each line in file if a variable is repeated twice then the last value only will be taken into consideration**.**

*Example:*

*Contents of file:*

*A= 3*

*A=2*

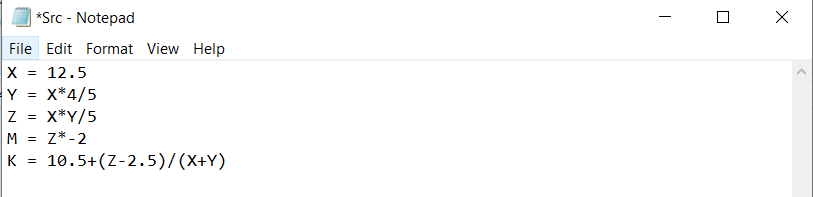
*Then variable A will only be stored once with value 2.*

Then, contents of file are printed without sorting. Then we add spaces to be sure the spaces inserted carefully and then if the expression is equation we change it from infix expression to postfix expression and evaluate the postfix expression. Afterwards, we create the binary search tree with each node containing variable symbol and their value then we print the variables and their values after sorting them by variable name (symbol) with in order traversal of binary search tree.

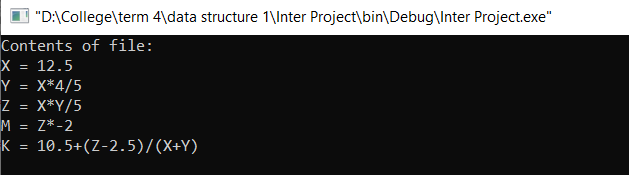
Afterwards, we create the heap after inserting all the values inside and array. Then, we sort the variables using variable values with a heap sort algorithm. Finally, the variables are again printed after sorting them with variable values.

**3- Sample Run:**

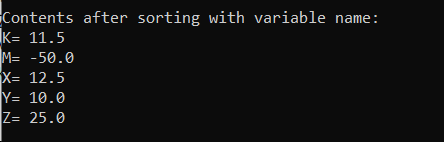
Contents of file:



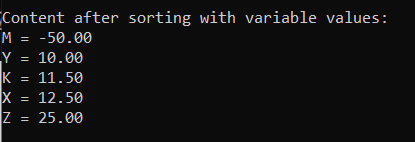
Firstly the program starts by reading everything inside the file:



Then, it calculates the value of each variable and prints them by sorting with variable name (sorting using in order traversal for binary search tree):



Lastly, it sorts the variables by their values (sorting using heap sort algorithm) and prints them:



Screenshot of the sample run as a whole:

