Report of assessment

Zihui Shi

May 25, 2022

Contents

1 Basic design for the application **1**
1.1 A description of the application . . . 1
1.2 Comment about the implementation of all the six tasks . . . 1

2 A description of the algorithmic choices you made for the application **2**
2.1 Justification of selecting and implementing specific Data Structures . . . 2
2.2 Description and justification of the Classes and Functions . . . 2
2.3 Describe and comment on unit tests . . . 3

3 Reference list **3**

1 Basic design for the application

1.1 A description of the application

This application contains two python files: 'assessment2', 'unit_test', and a txt file: 'readme'. Main code are in assessment2 file.

This appliction is used to culculate the expression like $((2+4)*5)$. It will culculate the result of the expression, and will print out the Tree of this expression if this expression is valid. If this expression is not vaild, it will return the reason why it is not valid.

An expression will be asked to be inputed, then it will changed into a list like ['(', '(', '2', '+', '4', ')', '*', '5', ')'].

Then we can save operators as internal nodes of a tree, and save numbers as leaves of the tree. In this tree, every subtree is a subexpression of the whole expression. These program use inorder traversal to culculate the result of the expression. These means the lower level expression has higher priority, and will be culculate first.

1.2 Comment about the implementation of all the six tasks

Task 1: the program can run, and it can firstly change the expression into a list, and will add the list's elements into a tree and use inorder traversal to culculate the equation, then print out the correct result.
Task 2: the program can visualise the generated binary tree with output in the terminal. Firstly set $p = 0$; When it meet element '(', $p + 1$; When it meet element ')', $p - 1$; When it meet a number, it will print p blanks before printing this number; When it meet a simbol in ['+', '-', '*', '/'], it will print (p-1) blanks before printing this simbol.
Task 3: Using Pickle package, the program can use pickle.dumps to save the binary tree into a file, and can also use pickle.loads to read the file back in.
Task 4: the program can report why the expression is not vaild. Three cases will be givien. If the expression has wrong number of operands, it will raise: "Not a valid expression, wrong number of operands". If the expression has operator missing, it will raise:"Not a valid expression, operator missing". If the brackets of the expression do not matched, it will raise:"Not a valid expression, brackets mismatched".
Task 5: the program use pickle package. Three unit test are included in the python file 'unit_test'. This file import 'assessment2' as a package. and test functions in 'assessment2' file.

Task 6: the code can be easy to read, it has self-describing method names. There is no dead code in this program, every code has its usages. It has annotation in the program, which can make it easier to understand. There is a Readme.txt in my code files. It explains how to run the program and the information about the code.

2 A description of the algorithmic choices you made for the application

2.1 Justification of selecting and implementing specific Data Structures

In thie program, BinaryTree and Stack data Structures are used. As Binary Tree is a kind of recursive data type, so we can use recursion to deal with the expression. We can use 'insert_left/right' to creat a letf/right subtree. We can use 'setRoot' to set value as the current node. We can use 'getLeft/RightChild' to set current node to the left of right subtree. We can use Stack to trace the parent node, when the current node moves down, it will 'push' the node into stack; when the current node moves up, it will 'pop' it out of the stack.

2.2 Description and justification of the Classes and Functions

(1) Three main Classes are used in this program.

Class 1: Stcak Class. There are four methods in this class: 'push', 'pop', 'size', 'is_empty'. 'push' method receives a new element 'a', and add a to this stack. 'pop' method will delete a element if the stack is not empty, otherwise it will raise 'the stack is empty'. 'size' method return the length of this stack. 'is_empty' method will return False if the stack is not empty, and True if it is empty.

Class 2: BinaryTree Class. There are six methods in this class: 'insert_right', 'insert_left', 'getRightChild', 'getLeftChild', 'setRoot', 'getRoot'. 'insert_right' method receives a new element 'new_data', and will insert it to the right child place of the current root. 'insert_left' method is generally the same as 'insert_right'. 'getRightChild' method will return the right child of the current root. 'getLeftChild' method is generally the same as 'getRightChild'. 'setRoot' method receives a value and set this value as the current root. 'getRoot' method returns the current root.

Class 3: Call_func Class. There is only one method in this class: 'call'. This method is used to call functions and Classes in order to run the code. This method will first ask user to input an expression, then change the expression into a list. Then it will call 'is_valid' function and 'is_matched' function to see wether the expressionis valid. If the expression is valid, it will call build_tree' function to form the tree. Then it will call 'evaluate' function to calculate the result of the expression. Then it will call 'print_exampleTree' function to print the tree out.

(2) Five main Functions are used in this program.

Function 1: build_tree function. It is used to build a tree of the input expression. It call BinaryTree Class and Stack Class. When we meet '(' , we will creat a new left child, and set this left child as the current root. When we meet a element in ['+', '-', '*', '/'] we will set this simbol as current root, and creat a new right child, and set this right child as the current root. When we meet a number, we will set this number as the value of current node, and back to the parent node. When we meet ')', we will back to the parent node of current node. If the case we meet is not above, it will raise 'ValueError'.

Function 2: evaluate function. It is used to calculate the result of the expression. It use operator package. It use recursion to do the culculation.

Function 3: print_exampleTree function. It is used to visualise the generated binary tree. Firstly set $p = 0$; When it meet element '(', p will plus 1; When it meet element ')', p will minus 1; When it meet a number, it will print p blanks before printing this number; When it meet a simbol in ['+', '-', '*', '/'], it will print (p-1) blanks before printing this simbol.

Function 4: is_matched function: It is used to determine whether brackets are matched. When the brackets are not matched, which means the number of '(' and ')' are not the same, it will return 'Not a valid expression, brackets mismatched'; when the brackets are matched, it will return nothing.

Function 5: is_valid function: It is used to determine wether the expression is valid. There are two cases of invalid expression, one is 'Not a valid expression, wrong number of operands', another is 'Not a valid expression, operator missing'.

2.3 Describe and comment on unit tests

There is a python file called 'unit_test'. It uses unittest package, and contains a Class: 'uni_test'.

There are three unit test in this class. These unit test respectively test 'print_exampleTree', 'is_matched', 'is_valid' three functions. If functions run, tests will pass.

3 Reference list

[1] JC1503-week8-slides-page-16
[2]
https://blog.csdn.net/weixin_44704985/article/details/108517154?utm_source=app&app_version=5.4.0&code=app_1562916241&ul