Programming Languages and Compilers

Final Exam

Please complete the exam and submit it as a plain text email with the subject "CISC 7120 Final" to nzhou@brooklyn.cuny.edu by midnight on Wednesday, December 16.

Question 1

Give a regular expression or a CFG for each of the following languages over $\Sigma = \{0, 1\}$.

1. Strings that contain no consecutive 1s.
2. Strings that contain an even number of 0s.
3. Strings that contain an equal number of 0s and 1s.
4. $\{0^n 1^{2n}\}$: n 0's followed by double the number of 1's.
5. Strings that are 2's power as binary numbers. For example, 0, 1, 10 and 100 are valid, but 11 and 101 are not.

Question 2

An identifier consists of letters, digits, underscores (_), and dollar signs ($), but cannot begin with a digit.

- Give a regular expression for identifiers.
- Write a function in a language of your choice that takes a string, and returns true if the string is a valid identifier and false otherwise.

Question 3

Consider the following CFG for postfix expressions.

```
E -> E E + | E E - | E - | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
```

(a) Is the grammar ambiguous? Why?

(b) Transform the grammar into one that does not contain left recursion.

(c) Transform the grammar of (b) into one that does not contain common prefix expressions on the right-hand side of each of the productions.

(d) Implement an evaluator in a programming language of your choice for the grammar of (c) based on the recursive-descent parsing framework. The top function takes an expression as a string and returns the evaluated value. For example, for `"32-4+"`, it returns `5`. If the string is not a valid expression, the function throws an exception.

Question 4

Write each of the following function in Picat, Haskell or Python:

1. `sorted_down(lst)`: This function checks if `lst` is sorted in *non-increasing* order. For example, for `lst = [3,3,2,1]`, it succeeds.
2. `triplets(lst)`: This function splits `lst` into 3-element groups from left to right, and returns a list of such groups. If the size of `lst` is not a multiple of 3, then the last group contains fewer than 3 elements. For example, for `lst = [a,b,c,d,e,f,g]`, the returned list is `[[a,b,c],[d,e,f],[g]]`.
3. `abc(lst)`: This function takes a list, and return true if the string matches the pattern $a^n b^n c^n$ and false otherwise. For example, for `lst = [a,a,a,b,b,b,c,c,c]`, it returns true.
4. `gen(int n)`: This function returns a list of all possible strings of that contain the same number of a's as b's. For example, for `n = 3`, the returned list is empty, and for `n = 4`, the returned list is `["aabb", "abab", "abba", "baab", "baba", "bbaa"]`. The order of the strings in the list is not important. `["aaa","aab","aba","baa"]`.

Question 5

Design a data structure for binary trees and write each of the following functions on binary trees in a language of your choice:

1. `member(x,btree)`: This function checks if `x` occurs in `btree`. The binary tree `btree` is not necessarily a binary search tree.

2. `one_child_node_values(btree)`: This function takes a binary tree and returns a list of values of the nodes in the tree that have exactly one child. The order of the values in the list is not important.

3. `equal(t1,t2)`: tests if tree `t1` and tree `t2` are equal. Two binary trees are equal if (1) both are empty; or (2) the roots have the same value, the two left subtrees are equal, and the two right subtrees are equal.

4. `shallowest_leaf(btree)` (extra 5 points): This function returns the value in a shallowest leaf node in `btree`. If there are multiple such leaves, then the function returns the left-most one.