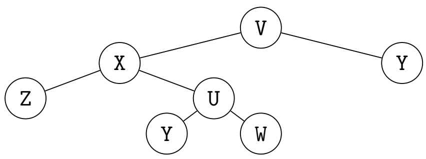
7 Algorithms (fms27)

Consider binary trees whose nodes have three fields: key (a single character between `U` and `Z`), left subtree, right subtree, with the two subtrees either both empty or both non-empty. Assuming suitable constructors, indicate an empty tree as `T()` and a non-empty tree as `T(key, leftSubtree, rightSubtree)`.

(*a*) Define unambiguous pre-order, in-order and post-order representations for such a tree `t`, called $\mathtt{r}_{pre}$`(t)`, $\mathtt{r}_{in}$`(t)`, $\mathtt{r}_{post}$`(t)`, consisting of strings over the {`U`, `V`, `W`, `X`, `Y`, `Z`, `(`, `)`} alphabet, with balanced brackets, with three characters (two brackets and a letter) for each node, starting and finishing with a bracket unless `t` is empty. Formally describe your three representations for a generic tree `t`, then produce the corresponding strings for the following tree. [6 marks]



(*b*) In this obfuscated pseudocode, the input `v0` is a syntactically correct $\mathtt{r}_{post}$`(t)`. Clearly explain (*i*) the purpose of the code; (*ii*) how it works; and (*iii*) how one should invoke it. Substitute meaningful explanatory identifiers for those $\mathtt{T}_x$, $\mathtt{v}_y$ and $\mathtt{m}_z$. Identifiers `v4` and `v6` are worth more marks than the others. [7 marks]

```
0  class T1:
1      # data members of objects of type T1
2      v2: T14 of T3
3      v4: T12
4      v6: T13
5
6      def m5(v0):
7          if v0 is "":
8              return T3()
9          else:
10             for v9 in v0:
11                 m10(v9)
12             return v2.pop()
13
14     def m10(v11):
15         if v11 is one of {"U", "V", "W", "X", "Y", "Z"}:
16             v6 = v4 is ")"
17         else if v11 is ")":
18             if v6:
19                 v8 = v2.pop(); v7 = v2.pop()
20             else:
21                 v8 = null; v7 = null
22             v2.push(T3(v4, v7, v8))
23         v4 = v11
```

(*c*) Explain in detail (*i*) how line 16 works, and (*ii*) why `v4` will never be uninitialised when line 16 is executed. [4 marks]

(*d*) Write clear pseudocode that takes a Tree `t` and returns $\mathtt{r}_{in}$`(t)`. [3 marks]