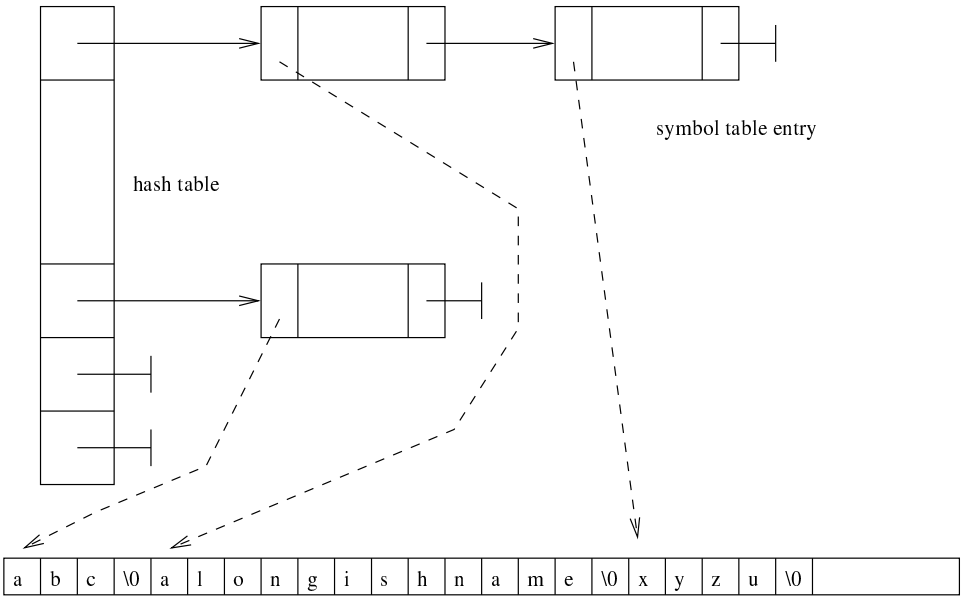

Figure 4.3: A simple symbol table using a string pool and hashing

```
1  float   x; /* def 1 */
2
3  int
4  f(int a)
5  {
6  while (a>0) {
7          double  x = a*a,i; /* def 2 */
8
9          for (i=0;(i<a);++i) {
10                 int x = a+i,b;  /* def 3 */
11                 b = x+2;        /* refers to def 3 */
12                 }
13
14         }
15 }
```

In the above examples, the symbol table will contain three entries (definitions) for x at the time the inner loop is processed. Which definition is meant is determined using the "most closely nested scope" rule.

In a compiler for a language with scope rules, when a semantic action wants to retrieve the information associated with a certain occurrence of a name, it has to know which definition of the name corresponds to the given occurrence. If the language allows only nested scopes for which the "most closely nested scope" rule holds, this can be solved by replacing the symbol table by a stack of simple