CS 61: Systems Programming and Machine Organization

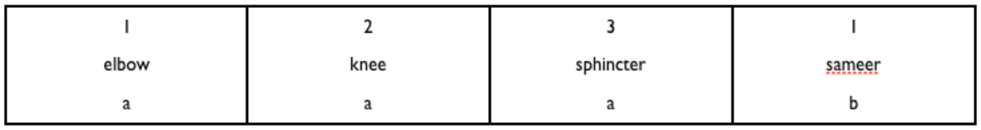
Topic: Automatic and semi-automatic memory management — Lecture #8
Scribes: Albert Wu, Ben Shryock, Vladimir Bok — 09/20/2012

0.1 Dictionary Example

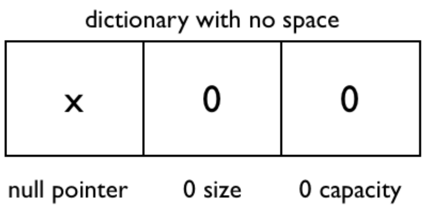
Given $n \geq 1$ dictionary files and $m \geq 1$ words, for each word, report in which dictionary it is.

Implementation:

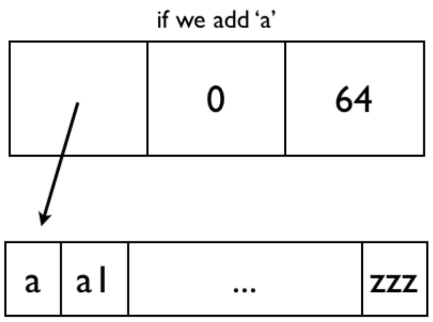
```
struct dict_word {
    char* word;
    char* file;
    int line;
}
```



The `line`s are allocated contiguously in memory. The `word`s and the `file`s are not, but the letters within the word and the letters within the file are (Note: this is why in the first lecture, the linked list implementation, which was non-contiguously allocated, was much slower than the array implementation, which was contiguously allocated).



A way to create an enlargeable data structure (like a dictionary) is to start by dynamically allocating a certain amount of space. When we exceed capacity, double the size by dynamically allocating more.



0.2 Bug

Consider a dictionary with $m = 10$ words: "one," "two," ... "nine." Run the function `dictionary_lookup`, which looks through all the words in the dictionary and compares each word with the word we are looking for. However, something is wrong: when we run the program to look for "zero", "one", "two", etc., the output is not correct. It reports that "nine" is on line 1. When we print out the word pointers, they all point to the same area of the stack since we assign the word pointer to a local variable "buf". So, since "nine" is the last value of this local variable, all of the word pointers are at that point of the stack.

0.3 Smashing the Stack

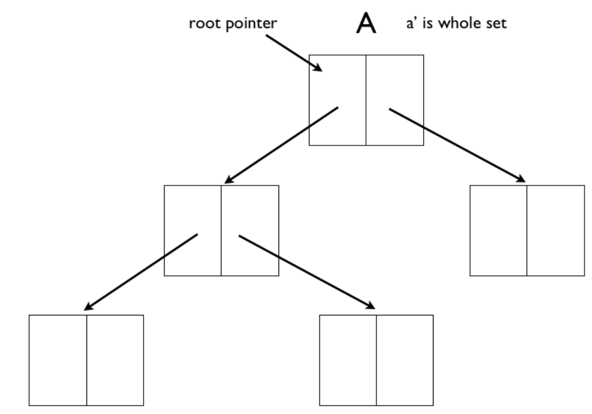
A stack buffer overflow (also known as stack smashing) occurs when a program writes to a memory address on the program's call stack outside of the intended data structure; usually a fixed length buffer (definition from Wikipedia, 2012).

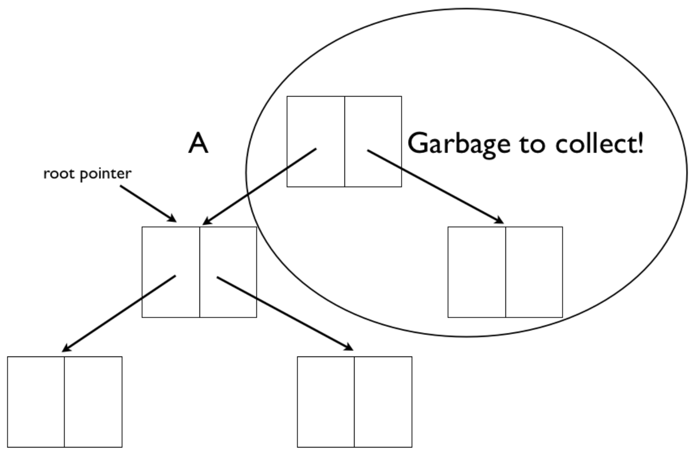
0.4 Memory Leaks

Memory leaks are when we allocate memory without freeing it. C has explicit dynamic memory allocation, i.e. user calls `malloc()` and `free()`. Other languages, such as Java, Lisp, Perl, or Ruby have implicit dynamic memory allocation with garbage collection.

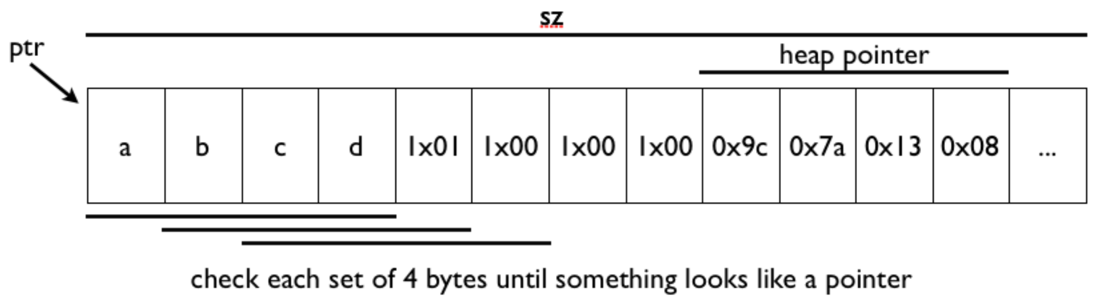
0.5 Garbage Collection

In computer science, garbage collection (GC) is a form of automatic memory management. The garbage collector, or just collector, attempts to reclaim garbage, or memory occupied by objects that are no longer in use by the program. (definition from Wikipedia, 2012). Given a set of allocations A and one or more root pointers $R \subset A$, find the smallest set A' with $R \subset A' \subset A$ so that all pointers in A' point within A.





Writing a garbage collector in C (need a set of roots, being able to chase pointers). First, we need to find roots for stack (bottom is the address of stack in main). Second, create a find function. There is a `ptr` at the beginning of memory at the beginning of a block of size `sz`. Pointers look like integers except they are in special ranges when pointing to the heap or the stack. When we get to a set of 4 bytes, we check the pointer.


sz
ptr
heap pointer
a
b
c
d
1x01
1x00
1x00
1x00
0x9c
0x7a
0x13
0x08
...
check each set of 4 bytes until something looks like a pointer

0.6 Tip For Assignment 1

Track memory allocations by defining a `struct memregion` that stores the pointer and size of the allocated region.