Name: Eland Anthony

CS403/503 Programming Languages
Spring 2021
Assignment #2

1. Problem 7 of Chapter 5 on Page 229:

Answer:

Under static scoping, x is 5. Under dynamic scoping, x is 10.

2. Problem 10 of Chapter 5 on Page 231

Answer:

At mark one, the visible variables, or variables within scope are a[from definition 1], and b, c, d[from definition 2]. At mark 2, e is also declared within the same scope, so a[from definition 1], b [from definition 2], c, d, and e[from definition 3], are the visible variables. At mark three, we close our while loop in which e was defined – therefore we have a[from definition 1] and b, c, d[from definition 2] as visible variables. At mark four, we close another while loop and definition 2 is disregarded from our scope, so we have only a, b, and c[from definition 1] as our visible variables. Variables not described for each mark are invisible.

3. Problem 11 (b-f) of Chapter 5 on Page 231

Answer: Assuming dynamic scoping is used here:

a) –
b) main calls fun1; fun1 calls fun3.

We have d, e, and f visible in fun3, and b and c only visible in fun1, while a is visible in main

c) main calls fun2; fun2 calls fun3; fun3 calls fun1.

We have b, c, and d visible in fun1, and e and f visible in fun3, and , while a is visible in main

d) main calls fun3; fun3 calls fun1.

We have b, c, and d only visible in fun1, and e and f visible in fun3, while a is visible in main

Name: Eland Anthony

e) main calls fun1; fun1 calls fun3; fun3 calls fun2.

We have c, d, and e in fun2, f in fun3, b in fun1, and a in main

f) main calls fun3; fun3 calls fun2; fun2 calls fun1.

We have b, c, and d in fun1, e in fun2, f in fun3, and a in main

4. Review Question 37 of Chapter 6 on Page 296

Answer:

The eager approach to reclaiming garbage utilizes reference counters. Reference counters maintain a counter that stores the number of pointers that are currently pointing at the cell for each cell. When a pointer is disconnected from a cell, a decrement operation occur within the reference pointer and the value is checked for whether or not the value is zero. If it is zero, the cell is identified as garbage, and in turn returned to the list of available space.

The lazy approach to reclaiming garbage, also known as the mark-sweep operation first allows garbage to accumulate as it allocates storage cells and deallocates pointers from the cells as needed. The mark-sweep operation then triggers – with the help of an extra indicator bit existing in each cell. Once this bit is set, the mark sweep operation allocates each pointer of each cell in the heap, and marks reachable cells as not garbage. It then enters the final phase, also known as the sweep phase, in which each cell that was not marked as ‘not garbage’ is returned to the list of available space.

Name: Eland Anthony

5. Problem 4 of Chapter 6 on Page 297

Answer:

	Tombstone method	Lock and Key method
Safety	Tombstone safely deallocates pointers by being assigned invalid values and prevents a pointer from ever pointing to a deallocated variable. Tombstones are inherently safe as they can accurately depict whether a pointer is allocated whilst utilizing indirection.	Lock and Key is generally safe, but sacrifices some safety for functionality and speed, for example: any number of pointers can reference a given heap-dynamic variable as only the lock value is changed. Thus, each copy will retain its address; however, the key value will not match the lock value so access will be denied.
Implementation Cost	Tombstones are never deallocated and must be allocated. – Negatively impacts space cost Accessing variable through tombstone requires extra machine cycle – Negatively impacts time cost	Memory must be allocated for the lock value in both the header cell and key value cell. – Negatively impacts space cost(minimal) Key and lock values must be compared – Negatively impacts time cost

Comparatively, the lock-and-key method is more suited for implementation cost, and the tombstone method for safety.