CS164 Programming Languages and Compilers **Spring 2024**

Discussion Worksheet 7

1 Reference Counting

Exercise 1 Consider the following Java program performing some linked list operations:

```
class Node {
    int value;
    Node next;
    void init(int value, Node next) {
        this.value = value;
        this.next = next;
    }
}
Node foo() {
    Node head = new Node(1);
    head.next = new Node(2);
    Node mid = head.next;
    mid.next = new Node(3);
    return head;
}
Node x = foo();
// line for part 1
_____________________   // line for part 2
x = null;
```

1.1 We want to perform reference counting garbage collection on the code. What is the reference count for each node object (designated by its value) at the line with the comment "line for part 1"?

`Node(1)`: _____ Solution: 1

`Node(2)`: _____ Solution: 1

`Node(3)`: _____ Solution: 1

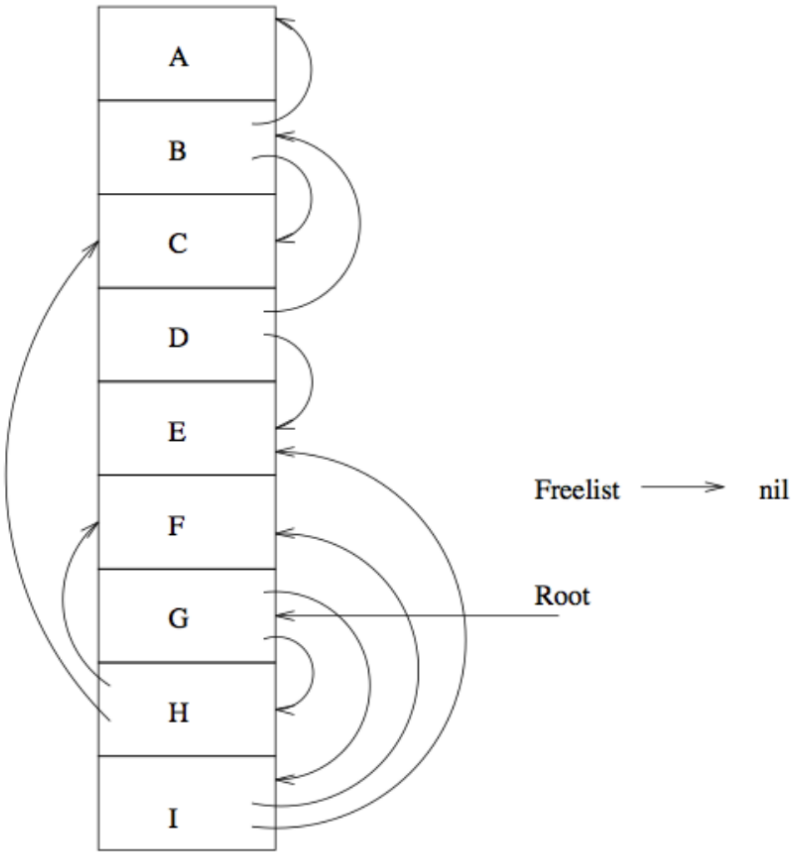
Solution: Note that `mid` goes out of scope when we return from `foo()`.

1.2 Fill in the blank line with code that would make reference counting fail to reclaim all memory when the program finishes. Do not add new variables or create new objects. Give the reference counts after this change when the program finishes, after GC is performed. What weakness of reference counting does this expose?

Solution: The code to add is `x.next = x`
This creates a circular reference inside x, since x has a field that refers to itself. Therefore, when the program finishes, the reference count of `Node(1)` is still 1. `Node(2)` no longer has a reference to it from `Node(1)`, making its reference count 0, so it will be collected by GC. `Node(3)` loses its reference from `Node(2)`, making its reference count 0 as well.

2 Mark & Sweep

Exercise 2 Garbage collect the following heap using Mark & Sweep garbage collection. Clearly indicate which cells will be marked, and construct the free list resulting from the collection.



Marked: _____ Solution: C, E, F, G, H, I

Freelist: _____ Solution: $A \to B \to D$

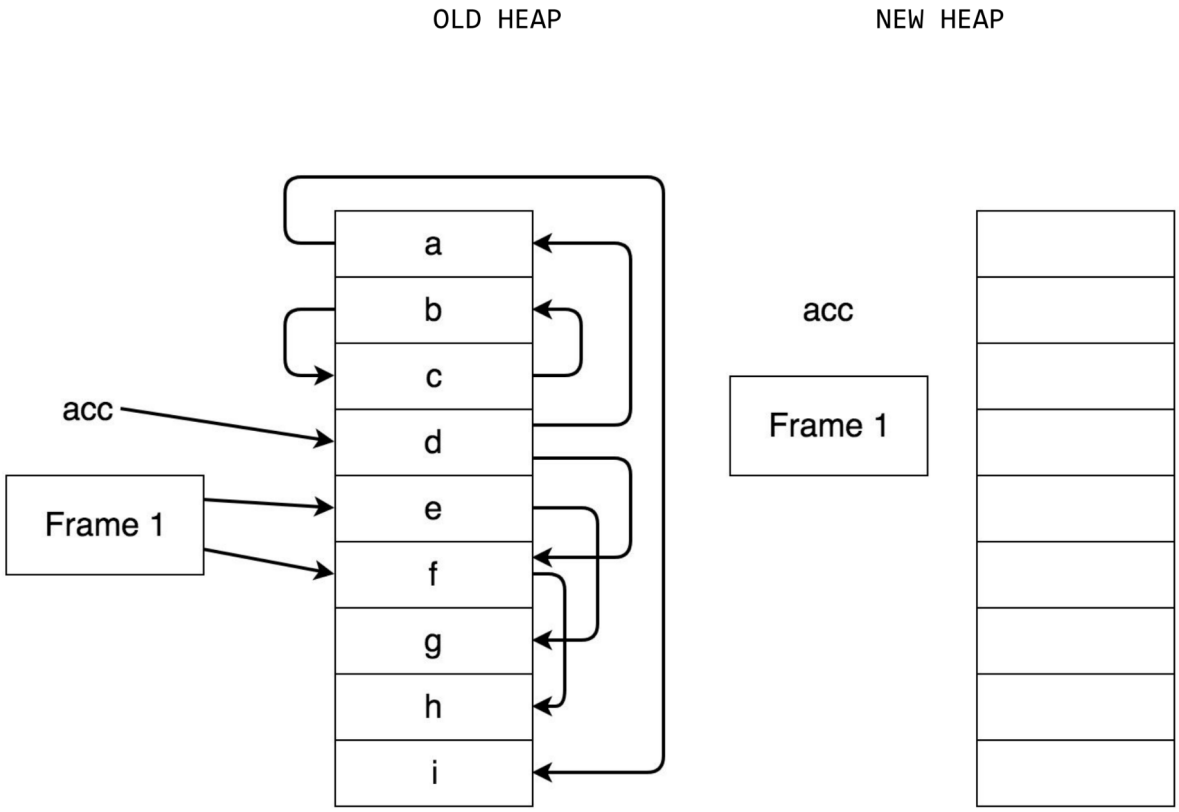
3 Stop & Copy

Exercise 3 Show the result of running the Stop and Copy garbage collection algorithm from class on the heap below. Fill in the objects and pointers in the new heap. You need not represent the forwarding pointers from the old to the new space.

Some important points:

- The roots are processed in the order: acc, Frame 1

- When an object has several pointers to other objects, process the pointers in alphabetical order of destination.



Solution: