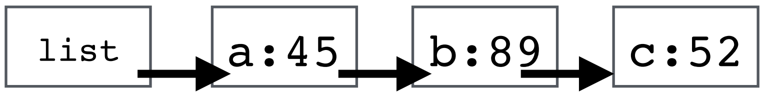
CSci 2113 - Lecture 5: From C to Java

Prof. T. Wood Fall 2016



1. Fill in the code and memory diagram so that the linked list will match the structure shown above. Assume ints and pointers take 4 bytes.

```
struct LNode {
  int data;
  struct LNode* next;
};
struct LList {
  struct LNode* head;
};

int main() {
 struct LList* list;
 struct LNode *a, *b, *c;
 list = NULL;
 a = NULL; b = NULL; c = NULL;
```

Stack

Address	*Name*	*Contents*
10000		
10004		
10008		
10012		
10016		

Heap

Address	*Alloc?*	*Contents*
50000		
49996		
49992		
49988		
49984		
49980		
49976		
49972		
49968		

CSci 2113 - Lecture 5: From C to Java

Prof. T. Wood Fall 2016

2. Write an algorithm (NOT code) which will append a new data node to a linked list. You can assume the linked list is structured similarly to the one in problem 1, but your algorithm should not be language specific.

```
Add node to end of a list:
 inputs: a List to add to
         the value of the new data to put into the list
```

Self-Quiz

1. How is a linked list different from an array? When might you use each data structure?
2. If the LList struct was modified to have a pointer to both the head and tail of the list, how would that affect your append function?
3. A doubly linked list has a pointer to the next element and a pointer to the previous element. What might that be useful for?
4. What are some differences between Java and C? How are they similar?
5. Given two C and Java programs that have identical functionality, which would you expect to use more heap memory and which would use more stack memory?