B2 • Discussion #12 • 2021-04-16

{{ Remind us @ 10:38 am EDT to stop so that you can take the quiz! }}

Announcements

- This Monday is a holiday so there are no classes. Wednesday follows a Monday schedule so we will have normal class/lab Wednesday
- Exam #3 next Friday! (One week!)
- Under "Exam 3 Review" folder, there's a set of problems "Exam 3 Sample Problems" that you can start working on
- We will have a practice exam in class next Wednesday.
- Study session Thursday 7-9 pm EDT (problems will be available earlier in the day)
- Final Project due the last day of class so you have lots of time to work on it. However, don't procrastinate!

Review Material

- Dynamic memory allocation
- Intro to Linked Lists: must know
  - how to initialize a LL (first element is always a special case)
  - how to traverse a LL to do something with every element
- Draw the damn boxes!!!

(*) Look at the GitHub for extra practice programs and supplemental DMA notes

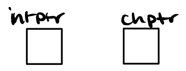
Example #1 (see GitHub for another program)

```
#include <stdio.h>
#include <stdlib.h>

void initthem(int **, char **);

int main()
{
    int *intptr;     } init./decl.    step ①

    char *chptr;

    initthem(&intptr, &chptr);     ← step ②

    printf("*intptr is %d and *chptr is %c\n", *intptr, *chptr);  ↰ step ④

    free(intptr);
    free(chptr);     ← step ⑤

    return 0;
}

void initthem(int **ipoint, char **chpoint)
{
    *ipoint = (int *) malloc(sizeof(int));
    **ipoint = 33;                                } step ③
    *chpoint = (char *) malloc(sizeof(int));
    **chpoint = '!';
}
```

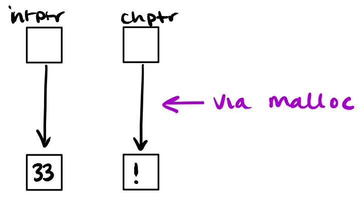
step 4:

*intptr is 33 and *chptr is !

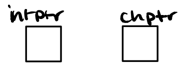
step 1:



step 2: & step 3:



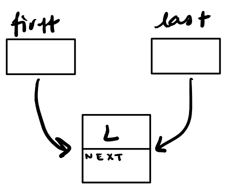
step 5:



Example #2 (see GitHub for Program)

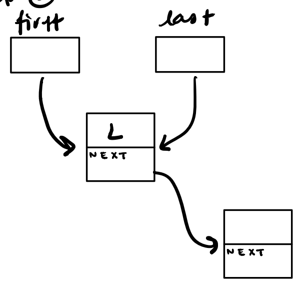
```
typedef struct linked_list
{                                         }
        char data;                        }
        struct linked_list *next;         }
}       element;                          }
typedef element *elementptr;

void trav_print(elementptr);   ← func prototype

int main()
{
        elementptr first = NULL,                    } initializations
                   last = NULL;                     }        +
        int i;                                      } declarations
        char myword[20] = "Lake Tahoe";             }

        first = (elementptr) malloc(sizeof(element));  }
        last = first;                                  } step ①
        first -> data = myword[0];                     }

        for (i = 1; i < strlen(myword); i++)
        {                                                          }
        (2a) last -> next = (elementptr) malloc(sizeof(element));  } step ②
        (2b) last = last -> next;                                  }
        (2c) last -> data = myword[i];                             }
        }

        last -> next = NULL;   } step ③
        trav_print(first);     }

        free(first);   } step ④
        free(last);    }

        return 0;
}
void trav_print(elementptr fir)
{
        elementptr current;

        current = fir;
        if (current == NULL)
                printf("There is no linked list!\n");
        else
        {
                while (current != NULL)
                {
                        printf("The data value is %c\n", current -> data);
                        current = current -> next;
                }
        }
}
```

Step ①:

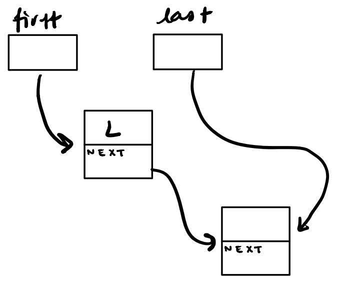


Step ②:

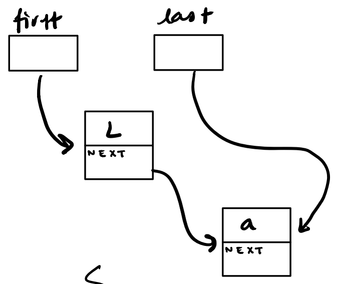
(2a)



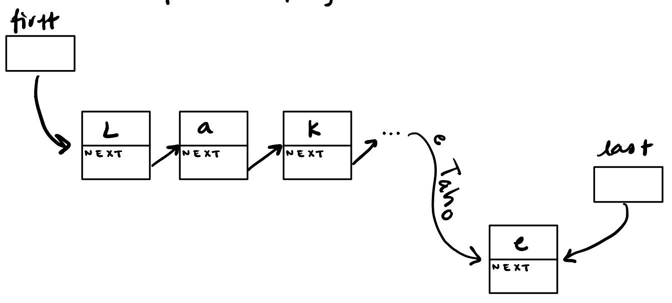
(2b)



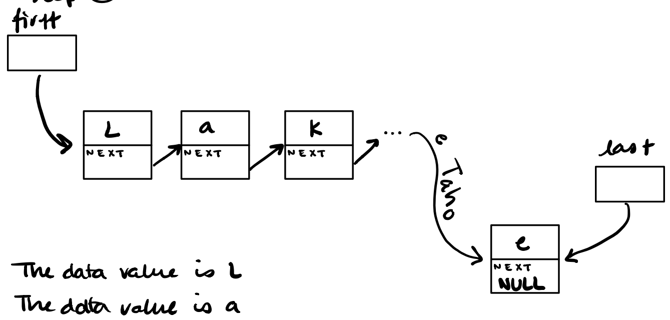
(2c)



keep on looping:

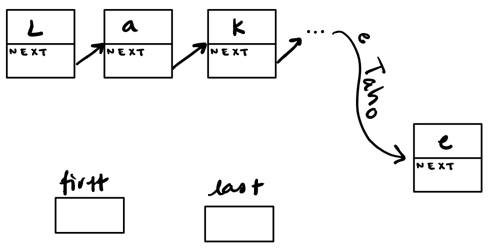


Step ③:



```
The data value is L
The data value is a
        ⋮
The data value is e
```

Step ④:



L
NEXT
a
NEXT
k
NEXT
...
e Taho
e
NEXT
first
last