



SIMON FRASER UNIVERSITY  
ENGAGING THE WORLD

# CMPT 125 - Introduction to Computing Science and Programming II

## Linked-Lists

# Linked List - Recap

- Chain of separate elements
- Head points to the first element
- Tail points to the last element
- Each element has a data part which contains value and a pointer which points to the next node in the list

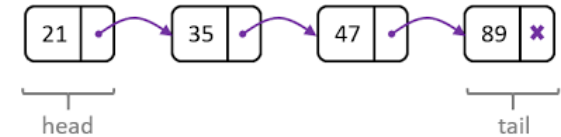


Fig1: Linked List  
Source: [101computing](https://www.101computing.com/linked-list/)

# Linked List - Operations

- **LL\_add\_to\_head(LL\_t \*list, int value):** Add element to the head of the list
- **LL\_add\_to\_tail(LL\_t \*list, int value):** Add element to the tail of the list
- **LL\_remove\_from\_head(LL\_t \*list):** Remove element from the head of the list
- **LL\_size(const LL\_t \*list):** Return the size of the list
- **LL\_print(const LL\_t \*list):** Prints all elements of the list from head to tail
- **LLnode\_free(node\_t \*node):** Frees memory used by the node
- **LL\_free(LL\_t \*list):** Frees memory used by the list

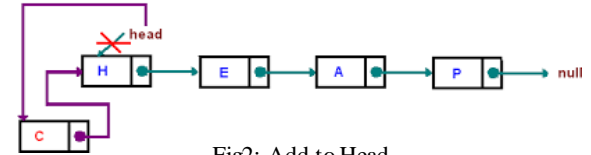


Fig2: Add to Head

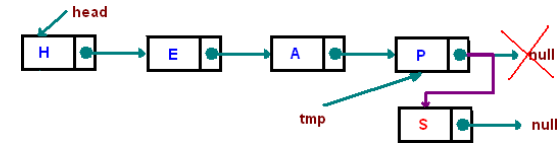


Fig3: Add to Tail

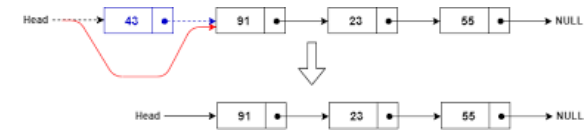


Fig4: Remove from Head

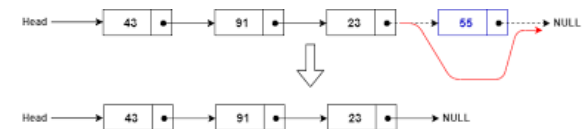


Fig5: Remove from Tail

# Steps to compile code

- Unzip and open the directory in VSCode

In the terminal, run:

- `> cd LL`
- `> make`
- `> ./driver_LL`

# Exercise

- Read and understand the functions defined in LL.c
- Implement the functions:
  - **LL\_remove\_from\_tail()**: removes element from tail of the list
  - **LL\_print\_reverse()**: prints list elements in reverse order. Try doing it in  $O(N)$  time.
  - **to\_array()**: gets a linked list and creates an array with same values
  - **array\_to\_list()**: gets an array and creates a linked list
  - **are\_equal()**: check if two linked lists are equal (equal length and same values in order)
- Add more test cases to test the functions you implement