



SIMON FRASER UNIVERSITY
ENGAGING THE WORLD

CMPT125 - Introduction to Computing Science and Programming II

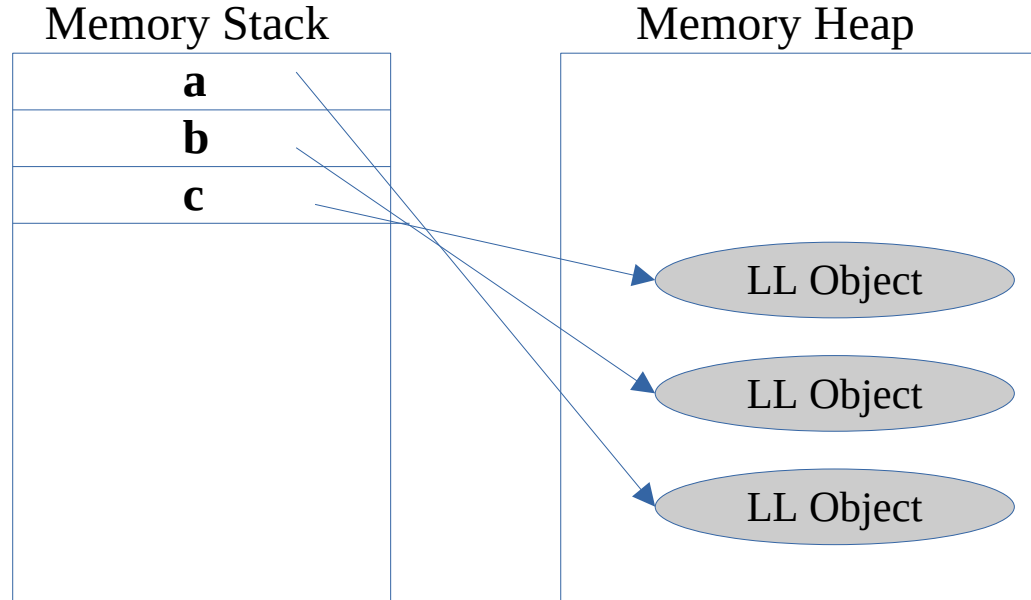
Quick Recap – Classes and Objects

```
LL* a = new LL();
```

```
LL* b = new LL();
```

```
LL* c = new LL();
```

Note that objects in heap are instances of class LL



- Aim: understanding Linked List and Stack
- To do:
 - Compile and run example4.cpp and example5.cpp
 - Read example4.cpp and example5.cpp and understand what each line is doing
 - open LinkedList folder and complete implementation for following functions in LL.cpp and LLnode.hpp
 - `LLnode(int value, LLnode* n) {}`
 - `void add_to_head(int value) {}`
 - `int remove_from_head() {}`
 - `LL::~~LL() {}`
 - open Stack folder and complete implementation for following functions in stack.cpp
 - `void stack::push(int item) {}`
 - `int stack::pop() {}`
 - `int stack::isEmpty() {}`
 - `stack::~~stack() {}`

Compile and run example4.cpp and example5.cpp

- compile them using g++

ex: "g++ example4.cpp" -- creates a.out and then run ./a.out

 "g++ example4.cpp -o run_ex4" -- creates run_ex4 then run ./run_ex4

- Please do similar thing for example5.cpp

- Open `example4.cpp`
- In the main function each time Constructors/Destructors are being called, in each print function please first try to guess what is going to be printed and why and compare your result with what has been printed.
- Please do same for `example5.cpp`

- Go to LinkedList folder.
- open LL.hpp, LL.cpp and LLnode.hpp.
- Complete implementation for following functions in LL.cpp and LLnode.hpp

- `LLnode(int value, LLnode* n){}`

it is in LLnode.hpp

It is the constructor that should take both value and next node, don't forget to update the head!

- `void add_to_head(int value){}`

It is going to add a note to the head with the data equal to value

- `int remove_from_head(){}`

It should delete first node in the list, don't forget to update the head !

- `LL::~~LL(){}`

Destructor. It should delete all nodes from the list before deleting itself.

- compile and test your code using make file.

- Go to Stack folder.
- open LL.hpp, LL.cpp and LLnode.hpp and stack.cpp and stack.hpp
- Complete implementation for following functions in stack.cpp

- `void stack::push(int item) {}`

It pushes a given item to the stack, we are assuming the stack is not empty.

hint: think about adding a node with a value to the head in linked list.

- `int stack::pop() {}`

It pops the element from the stack.

hint: thing about removing the element from the head in the linked list.

- `int stack::isEmpty() {}`

It check if the stack is empty.

hint: we had a getSize function in linked list and when the size is 0 it means it is empty.

- `stack::~~stack() {}`

Destructor