CMPT 225 - Call stack and heap memory

The call stack

A function A calls another function B, which calls c
During execution, control passing from (the code for) A, to B, than to V
When execution of C ends, control must return to B and then to A

At each function call, the system records where control should return to by pushing an activation record on the call stack
The call stack also records all local variables, including the arguments to the function call.

- The argument takes up space in the call process memory

Dynamic memory or heap

variables declared in function are stored on the call stack
These variables

- are of fixed size
- are destroyed when the function they are defined in terminates

We often want a function f to create data that can be used after f returns
In particular, dynamic data structures require this

This data is stored in the "heap" a region of memory that is allocated dynmically as needed.

A basic or primitive type can be stored on the call stack or on the heap
objects can also be stored on the call stack or on the heap

Variables declared in functions are on the stack
Allocation on the heap is denoted by "new"