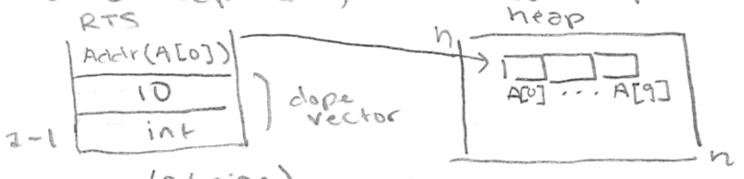
PLD | ch. 11 | Kurt Medley

Static memory contains values whose storage requirements are known before run time and remain constant throughout the life of a running program.
Run-time stack is the center of control for dispatching active functions, locally-declared variables, and parameter-argument linkage.
Heap contains values that are dynamically allocated and structured while the program is running, such as strings, dynamic arrays, objects.

new and delete - obtain and release a contiguous block of memory words in the heap.
↪ individual blocks of heap space are addressed using pointer variables, whose values are addresses.

Dynamic array allocation - The stack holds a reference to the address of a heap block, while the heap holds the arrays entries.



new(ad.size)
Call - push addr(ad[0]) onto stack; push ad.size onto the stack; push ad.type onto the stack.
→ if "new" does not succeed, heap overflow occurs
→ "delete" - clite returns control to the caller, any dynamically allocated array for that function must be deallocated by restoring all entries in its heap block to the unused state.

11.3 Identify all of the possible run-time errors that can occur for the array declaration or reference A[n] when it is encountered during the interpretation of a program.

dereferenced arrays can become orphans (garbage) when their presence is allocated by the user and then eventually set to some other variable array. This can cause run-time errors (stack overflow) and generally clogs up memory. The same can be said for dangling references.

index-out-of-bound errors can occur from the mismanagement of array sizes and their manipulation

Heapoverflow can occur when the heap is unable to support a call "new"'s allocation of a contiguous block of memory.

11.4. Array indices in Clite must be checked at runtime, since the size of an array cannot be statically determined. Use the indexing errors that you identified in the previous question, along with the heap-based model for array allocation discussed in this chapter as a basis for your expanded rule definitions.

* Compute addr (ad[0]) = new(ad.size), where the value of the Expression ad.size is computed as described in chapter 5.
* Push addr (ad[0]) onto the stack
* Push ad.size onto the stack
* Push ad.type onto stack.
→ if the size of the array pushed onto the stack exceeds the stack, throw a runtime stack overflow error.
→ throw an index out of range error if Meaning Rule 11.2 or Meaning rule 11.3 are violated.
→ Throw a heap overflow error if the heap is unable to support a "new" allocation of a contiguous block of memory.

11.5 Expand the meaning rule for a Call in Clite to incorporate the idea of heap memory allocation and recovery for an array parameter or local variable. This change will naturally utilize the new and delete functions.

Lexical and concrete Syntax of Call
Call → Identifier (Arguements)

AS
Call = String name; Expression args

Alter call to cover array references by
Call → String name; Expression | ArrayRef
if the call is an array reference, use the keyword new to allocate Meaning rule 11.1.

After a return statement that references a dynamically allocated array is executed, use "delete" to restore the values of the heap to "unused", so future heap manipulation can happen with reduced risk of heap overflow.