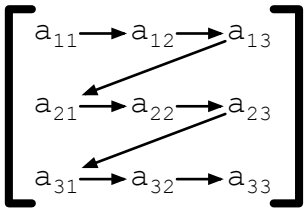
Lecture Notes 7

Data Types

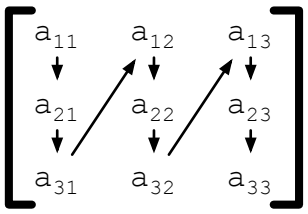
- Data Type – A collection of data values and a set of allowed operations on those values
- Descriptor – The collection of the attributes of a variable
- Primitive Data Types – Data type not defined in terms of another data type
- Derived Type – A type that is based on previously defined types
- Integer – Primitive data type that of varying sizes and ranges, typically stored in twos complement
- Float – Approximates a real number, most implement IEEE 754 standard
- Complex – Some languages support complex numbers as primitive values and will allow normal operations on them
- Decimal – Fixed number of decimal digits, often used in business to avoid roundoff issues of float, can be stored in Binary Coded Decimal (BCD)
- Boolean – Holds either true or false
- Character – Originally only represented ASCII, modern languages support UNICODE
- Character Strings – A sequence of characters
  - Substring Reference (or Slices) – Specify a range of a string that can often be interpreted as a string
  - Static Length String – String is fixed length and immutable
  - Limited Dynamic Length String – String can be modified and has a maximum length
  - Dynamic Length String – String can be modified to any size the system can support
- Enumerated Types – All possible values for the type are named and provided in the definition
- Arrays – Homogenous aggregate of data elements
  - Subscript (or Index) – Selector for element within array
  - Static Array – Storage allocation and index range is statically bound
  - Fixed Stack-Dynamic Array – Storage allocation is dynamically bound but index range is statically bound
  - Fixed Heap-Dynamic Array – Both storage allocation and index range are dynamically bound at allocation and fixed after
  - Heap-Dynamic Array – Storage allocation and index range can change during array lifetime
  - Rectangular Array – Multidimensional array where all rows have same number of elements
  - Jagged Array – One in which the rows don’t have to have the same number of elements

This content is protected and may not be shared, uploaded, or distributed.

- Row-Major Order – The method of storing a two-dimensional array in linear memory by having elements in the same row be contiguous in memory. Below is shows the logical traversal of Row-Major Order.



- Column-Major Order – The method of storing two-dimensional array in linear memory by having elements in the same column be contiguous in memory. Below shows the logical traversal of Column-Major Order



-

- Associative Arrays – Unordered collection of elements that are indexed by values called keys
  - Examples – Hashes in Perl, Dictionaries in Python
- Records (or Structs) – Aggregate of data elements where elements are accessed by names
  - Fully Qualified Reference – Full name of all intermediate record elements are included
  - Elliptical Reference – Part of the intermediate record element is omitted
- Tuples – Elements are unnamed but indexed and do not have to be homogenous like array
- Lists – A sequence that is indexed but unlike array the elements do not need to be homogenous
  - List Comprehension – Taken from set notation is a powerful mechanism to create a list
- Unions – Data type that allows same memory cell to be interpreted based upon the element name
  - Free Union – No mechanism to determine the type of value currently stored in the union
  - Discriminant (or Tag) – Specifies the type currently being stored in the union
  - Discriminated Union – A union with a discriminant
- Pointers and References – A variable that contains a memory address
  - NULL, null, or nil – An invalid memory address
  - Heap-Dynamic Variables – Variables allocated on the heap
  - Dereferencing – Act of accessing or updating the value of the variable that the pointer references

This content is protected and may not be shared, uploaded, or distributed.

- Alias – Two different names are bound to the same object at once

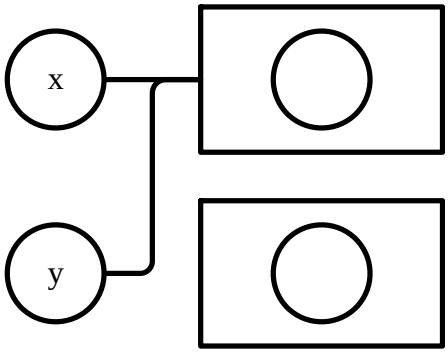


Figure 1. Alias with sharing

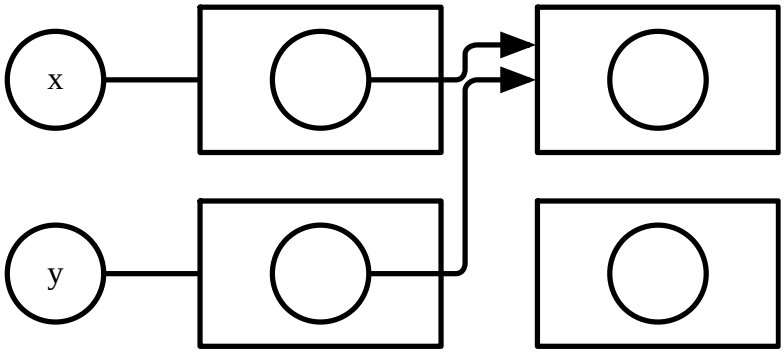


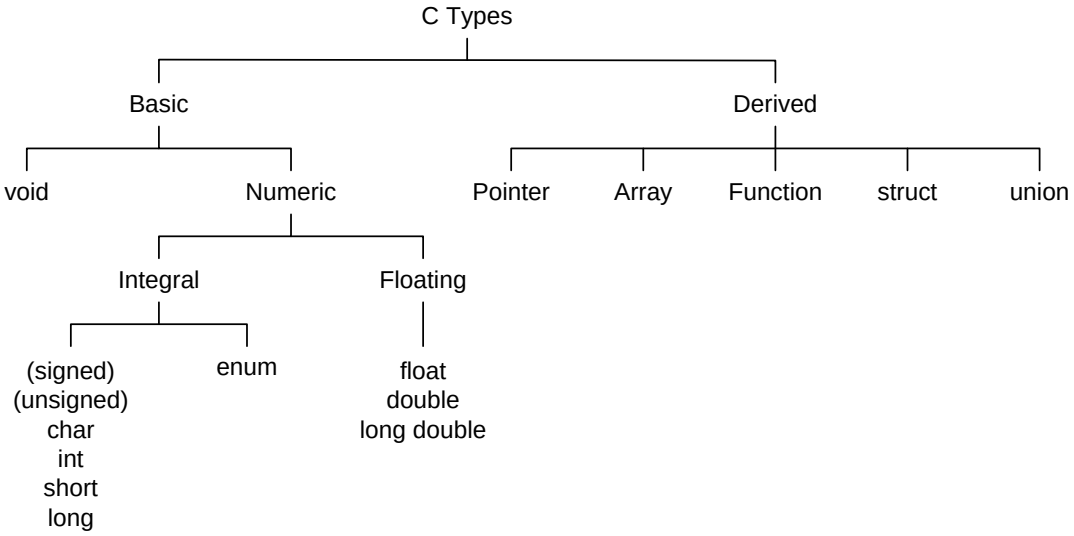
Figure 2. Alias with reference sharing

- Side Effects – Change of variable value that persists beyond the statement execution
- Dangling References – Deallocated location that still be referenced
  - Can be avoided with removal of explicit deallocation
- Garbage – In accessible memory that is allocated to the environment
  - Garbage Collection – Automatic reclaiming of Garbage
  - Reclamation – Reclaiming unusable memory for future use
  - Maintenance – Maintaining of the free space
  - Coalescing – Connect contiguous freed memory blocks into larger blocks
  - Fragmentation – Memory broken into small blocks
  - Compaction – Moving all free blocks together to coalesce into one large block
  - Reference Counting – Eagerly frees memory when reference count is zero
  - Mark and Sweep – Marks referenceable memory and sweeps all unmarked into free space
  - Stop and Copy (Reclamation) – Copy reachable objects into other half of memory, no need for sweep (can use half of total memory)
  - Generational Garbage Collection (Stop and Copy plus Permanent Storage) – Long living objects are copied to permanent storage

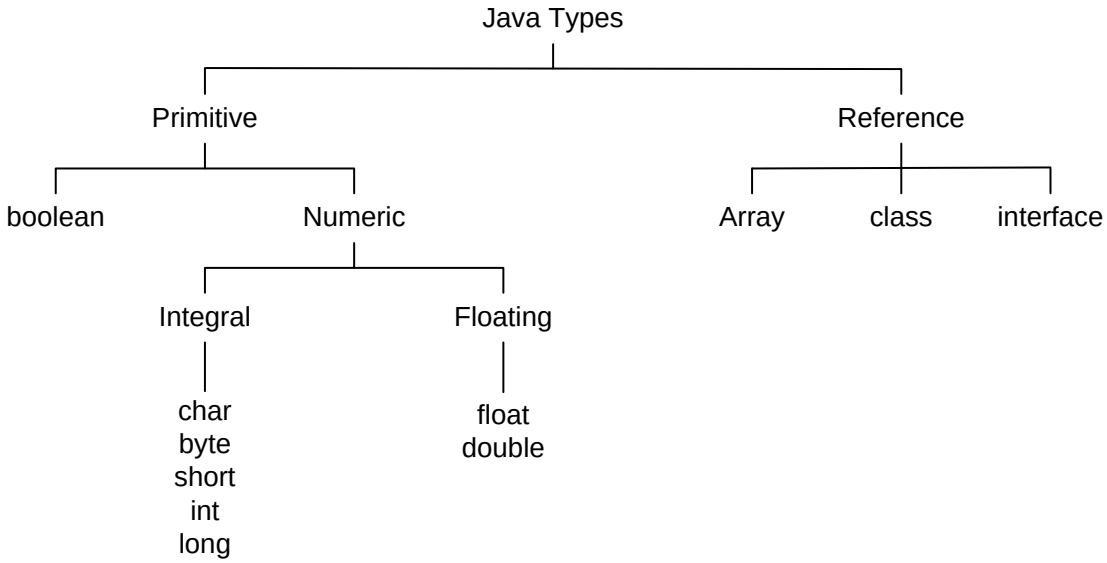
This content is protected and may not be shared, uploaded, or distributed.

- Reference Type – A variable type that references another object or value, not a specific memory address. Referenced variables can be moved in memory.

- Type Nomenclature
  - C

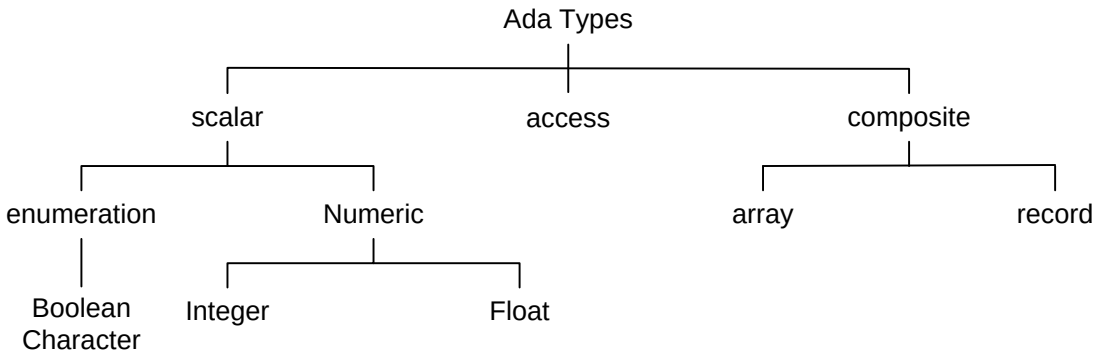


  - Java



This content is protected and may not be shared, uploaded, or distributed.

- Ada



This content is protected and may not be shared, uploaded, or distributed.