Vertex & Edge descriptors:

* manipulation of vertices and edges
* accessible via graph\_traits

Property maps: ch 15

* objects attached – properties
* get(p\_map,key)
* put(p\_map,key,value)
* p\_map[key]
* p=property
* ex: name\_map, delay\_map

Graph traversal:

* vertex iterator: traverse all vertices – value type of vertex iterator is vertex descriptor
* edge iterator: traverse all edges – value type of edge iterator is edge descriptor
* adjacency iterator: traverse all vertices adjacent to given vertex

Structure: user defined data types which stores group of non-similar data types

* data members
* member functions

Structure vs class:

* Structure is not secure
* Default in class is private

Polymorphism:

* Multiple forms
* Function overloading – signature different same name, operator overloading – ‘+’ can be add integers or concatenate strings, virtual functions

Pointers:

* Store address of memory location
* Datatype \*var\_name;
* \* before var while initializing denote its pointer. This variable stores address of another variable
* \*var\_name gives value at that address. Called dereferencing

Operators:

&:ampersand

* Int& ra = 10; reference to is declared. That is ra is reference to an integer and assigned number
* & mean address of
* &x gives address of x variable

\*:

* Pointer to