Name: Soham Thaker SID: 011-748-159 Email: [sdthaker1@myseneca.ca](mailto:sdthaker1@myseneca.ca)

Raw pointers are pointers that are the built-in types provided by the language. They are allocated on stack, however the memory that they point to can be either static or dynamic. If the memory they point to is statically allocated then the language guarantees that, that resource will be deallocated when the scope where its declared ends. However, if the resource is dynamically allocated then it is solely the programmer’s responsibility to deallocate the memory.

Smart pointers are user-defined types provided the STL. To use smart pointers, we must include memory header. Smart pointers are allocated statically but the resource they point to is usually dynamically allocated. They use the RAII principle which means that resource is acquired at the time of initialization. Smart pointers are of two types, namely unique pointers, and shared pointers. Both have a one argument constructor that acquires the resource at the time of initialization. Unique pointers by design only point to one resource at a time, which is why they have copy operations disabled. They can only be moved from one object to another whereas shared pointers share the same memory among multiple objects and need reference counting to determine when to deallocate the resource. They support copy and move operations.

One of the major advantages of smart pointers is they do not need to be deleted manually, since they are allocated on stack, and they will be deleted automatically, along with the acquired resource since once they reach end of life, their destructor is called which will delete the acquired resource. As we can see in this workshop, the mergeSmart function declares a local unique pointer object and when an exception is thrown or the object goes out of scope the destructor of unique pointer is called which in turn deletes the acquired resource, making the program leak free.

The difference between raw and smart pointer syntax is that raw pointer points to dynamically allocated memory which means it is not responsible to deallocate the memory whereas smart pointer acquires the object and is responsible to deallocate the memory, meaning it is in a composition relationship. As we can see in this workshop, in mergeRaw function programmer had to delete the memory manually causing leaks in situations when an exception is thrown whereas in mergeSmart function, once the object was out of scope it was destroyed properly, without causing any memory leak.