**Task 6 – 102564760**

Ryan Chessum

**Q.1 [line 102] There are inside array\_demo\_1 - answer them there.**

**Q.1.1 [line 164] What do the < and > mean or indicate?**

*<> is a template and is another way to initialise the array.*

**Q.1.2 [line 165] Why don't we need to write std:array here? (Is this good?)**

*Because we’re using namespace std at the top we can use things from the standard library without writing std:: before using them. This is great as we don’t need to write std:: before everything from the standard library so our code looks nicer, is a bit easier to read and saves us a bit of time.*

**Q.1.3 [line 166] Explain what the int and 3 indicate in this case?**

*In this case int is the data type for the items in the array and 3 is the number of elements in the array.*

**Q.1.4 [line 204] In the code above, what is the type of itr2?**

*Auto is a keyword which lets the compiler work out the type for us. If we use a break point we can see that Itr2 is set to an Array iterator.*

Graphical user interface

Description automatically generated with medium confidence

**Q.1.5 [line 211] In the code above, what is the type of v?**

*V is a constant reference of an int.*

Graphical user interface, background pattern

Description automatically generated

**Q.1.6 [line 212] In the code above, what does the & mean in (auto &v : a1)**

*& is a constant reference to an object or variable. So, if we manipulate it within a function, it wont manipulate a copy, the variable will be changed even outside of the function. It’s similar to passing a pointer as essentially what is happening is we are manipulating data at the address of the variable.*

**Q.1.7 [line 220] Try this. Why does a1[3] work but at(3) does not?**

*My IDE does not let me do either but in theory a1[3] would let us look outside the array otherwise because it does not have any bounds checking at() performs bound checking for us.*

**Q.1.8 [line 233] auto is awesome. What is the actual type of v that it works out for us?**

*int array iterator.*

A picture containing graphical user interface

Description automatically generated

**Q.1.9 [line 240] auto is still awesome. What is the actual type of v here?**

*Constant reference to an int.*

Graphical user interface

Description automatically generated

**Q.1.10 [line 250] How would you do a forward (not reverse) sort?**

Text

Description automatically generated

**Q.2 [line 105] In array\_demo\_2, explain what a4(a1) does**

*This calls a copy constructor and makes a copy of a1.*

**Q.3 [line 108] No questions for array\_demo\_3, it's just a demo of Struct/Class use with array.**

Text

Description automatically generated

**Q.4 [line 111] How do we (what methods) add and remove items to a stack?**

*Push() adds an item to the top of a stack and pop() removes the item at the top of a stack.*

Q.5 [line 112] A stack has no no [] or at() method - why?

Q.6 [line 115] What is the difference between a stack.pop() and a queue.pop() ?

Q.7 [line 118] Can we access a list value using and int index? Explain.

Q.8 [line 119] Is there a reason to use a list instead of a vector?

Q.9 [line 122] Was max\_size and size the same? (Can they be different?)

Q.10 [line 123] Which ParticleClass constructor was called?

Q.11 [line 124] Were the ParticleClass instances deleted? If so, how?

Q.12 [line 125] Was the vector instance deleted? If so, how do you know this?

Q.13 [line 126] Your IDE might suggest to use emplace\_back instead of push\_back. What does this mean?