# Comp 4603

# Advanced C++

|  |  |
| --- | --- |
| Assignment | 10 |

Student Name: Alisher Shamayev

BCIT ID: A01182685

Task 1: Elaborate profiling tools using your own words.

|  |
| --- |
| Profiling tools are helping the users with analyzing performance of the application and optimize code to get better level of performance. |

Task 2: Elaborate C++ reusable containers

|  |
| --- |
| Reusable containers are usually containers that are holding large amount of data which will be a same type of data. They also mostly flexible and generic, provide easy access to the data that being stored and works with big amount of data efficiently.  Examples of reusable containers are: vector,stack,list,set,queue,map. |

Task 3: Elaborate commonly used C++ reusable containers methods

|  |
| --- |
| Two of the most common methods are size() and empty() which used for first ones goal is to return the size of container, while second ones is to check if container is empty.  Most of them but not everyone also have method named clear() which makes the container to be cleared and empty.  Last common methods are pop() and pop\_back() or pop\_front()  So all of them are removing element from the list however pop and pop\_front removes top element, pop\_back removes last element. On the other there are similar push() and push\_front() or push\_back() which works same way but they add element instead of removing. Similar to this ones will be function insert() ads and erase() removes elements this one are for map and sets. |

Task 4: Elaborate C++ creational design patterns

|  |
| --- |
| Creational patterns in C++ are factory method, builder method, prototype, singleton and abstract factory method.  This Designs pattern is a design where the system is going through whole process of object creation from beginning until the end, and it have all the information of an object such as its creation, composition and representation. |

Elaborate C++ structural design patterns

|  |
| --- |
| Structural design patterns are Adapter, Bridge, Composite, Decorator, Façade, Flyweight and Proxy.  This design pattern is helping us to combine different objects or re structure them, so in short giving us new functionality. Overall we can say that this class helps us to combine or organize different classes or objects in one bigger structure with better function. |

Elaborate C++ behavioural design patterns

|  |
| --- |
| Behavioral design patterns are Chain of responsibility, Visitor, Command, Iterator, Null Object, Observer, Mediator and State.  Basically this pattern it relates and works with changing the behavior of an object so changes the algorithm or assignment of an objects or between the multiple objects as well. |

Task 5: Elaborate C++ architectural patterns

|  |
| --- |
| Architectural design patterns are Pipes and Filters, Layers, MVC, Reactor and etc.  Overall, this design pattern that will explain in detail or describe in details objects or classes fundamental structure. |