Question 1 (40%, each of the 3 sub-questions below are worth 10%,10% and 20% respectively)**.**

1. The Fibonacci sequence $a(1), a(2), a(3), \ldots, a(n), \ldots\ldots$ is defined by

$$a(1) = 1$$
$$a(2) = 1$$
$$a(n) = a(n-1) + a(n-2), \text{ for all n } > 2.$$

   This generates the sequence

$$1, 1, 2, 3, 5, 8, 13, 21, ...$$

   Write a C++ function "fibonacci(...)" that computes the Fibonacci number corresponding to its positive integer argument, so that, for example, fibonacci(7) == 13.

2. What is the difference between a reference and a pointer?

3. Predict the output of the following program. You should justify your answers.

```
#include <iostream>
using namespace std;

int main()
{
    int * ptr_a;
    int * ptr_b;
    int **ptr_c;

    ptr_a = new int;
    *ptr_a = 3;
    ptr_b = ptr_a;
    cout << *ptr_a << " " << *ptr_b << "\n";

    ptr_b = new int;
    *ptr_b = 9;
    cout << *ptr_a << " " << *ptr_b << "\n";

    *ptr_b = *ptr_a;
```

```
    cout << *ptr_a << " " << *ptr_b << "\n";

    delete ptr_a;
    ptr_a = ptr_b;
    cout << *ptr_a << " " << *&*&*&*&*ptr_b << "\n";

    ptr_c = &ptr_a;
    cout << *ptr_c << " " << **ptr_c << "\n";

    return 0;
}
```

Question 2 (30%)**.**

1. What is a C++ template? When is a template a better solution than multiple classes?

2. What is a pure virtual function? Use an example to explain their use in object oriented design.

3. Consider the following C++ code segment.

```
#include <iostream>
using namespace std;

class Account
{
    private:
        int ID;
        double max, min;
    public:
     Account() // Function 1
        {
          ID = 105;
          max = 100;
          min = -100;
        }

        Account(int r_no, int min_new,int max_new) // Function 2
        {
            ID = r_no;
            min = min_new;
            max = max_new;
        }
};
```

   Which object-oriented design concept is illustrated through 'Function 1' and 'Function 2'?

Question 3 (30%)**.**
Show that the variance of the optimal Markowitz portfolio is given by,

$$\text{var}(m) = \frac{Am^2 - 2Bm + C}{AC - B^2},$$

where m is the target return, and

$$\begin{aligned} A &= 1^\top \Sigma^{-1} 1, \\ B &= \mu^\top \Sigma^{-1} 1 \\ C &= \mu^\top \Sigma^{-1} \mu \end{aligned}$$

where Σ is the covariance matrix, μ is the vector of expected returns and 1 is a vector of ones of appropriate dimension. Use the result above to show that the covariance of the minimum variance portfolio given by,

$$\min_m \ \text{var}(m)$$

with any other portfolio is $1/A$.
Hint: The function $\text{var}(m)$ is strongly convex and hence has a unique minimum.

Answer 1.

```
int fibonacci(int number)
{
   if (number < 1)
   {
      cout << "\n\nError - non-positive argument to 'fibonacci'\n";
      exit(1);
   }
   else if (number == 1 || number == 2)
      return 1;
   else
      return (fibonacci(number - 1) + fibonacci(number - 2));
}
```

1. 3 3

   3 9

   3 3

   3 3

   a2644 3

   ptr c is a pointer-to-a-pointer.

   aa2644 is the address held in ptr a and may vary from run to run and machine to machine

2. Pointers and References are two different concepts. A pointer in C++ is basically a variable that stores the memory address of another variable. While references act as an alias name for an existing variable that needs to be initialized at the time of declaration. This is not the case with pointers.

Answer 2.

1. Templates allow creating functions that are independent of data type (generic) and can take any data type as parameters and return value without having to overload the function with all the possible data types. When you are designing a generic class to contain or otherwise manage objects of other types, when the format and behavior of those other types are unimportant to their containment or management, and particularly when those other types are unknown (thus, the generality) to the designer of the container or manager class.

2. Class constructors and polymorphism.

3. A Pure Virtual Member Function is a member function in which the base class forces the derived classes to override. Normally this member function has no implementation. Pure virtual functions are equated to zero.

   ```
   class Shape { public: virtual void draw() = 0; };
   ```

   Base class that has a pure virtual function as its member can be termed as an "Abstract class" This class cannot be instantiated and it usually acts as a blueprint that has several sub-classes with further implementation.

Answer 3. The Lagrangian is given by,

$$L(x, \lambda, \gamma) = w^\top \Sigma w + \gamma(1^\top w - 1) + \lambda(\mu^\top w - m)$$

Differentiating with respect to w and using the two constraints gives the desired result. The intermediate steps give,

$$w^\star = \Sigma^{-1}(\lambda\mu + \gamma 1)$$

and after solving the resulting system we get,

$$\lambda = \frac{Am - B}{AC - B^2}$$

$$\gamma = \frac{C - Bm}{AC - B^2}$$

The minimum variance portfolio is,

$$\min_m \text{var}(m)$$

Differentiating we get,

$$\frac{d\text{var}(m)}{dm} = 2Am - 2Bm = 0$$

i.e. the minimum variance portfolio had mean $m_g = B/A$and the vector of weights for this portfolio is $w_g = \Sigma^{-1}1/A$. Let w be any other portfolio then

$$\text{cov}(w_g, w) = w_g^\top \Sigma w = \frac{1}{A}$$