

Task 2

1. Describe the principle of **polymorphism** and how it was used in Task 1.

In Task 1, MinMaxSummary and the AverageSummary are two different strategies of the SummaryStrategy class or, in other words, Summary Strategies have 2 different forms that are the MinMaxSummary (minmax strategy (local variable)) and the AverageSummary (averagestrategy(local variable)). And when calling

```
analyser.Strategy = minmaxstrategy;
```

or

```
analyser.Strategy = averagestrategy;
```

The polymorphism is implemented

2. Using an example, explain the principle of **abstraction**. In your answer, refer to how classes in OO programs are designed.

Abstraction is one of the 4 key concepts of object-oriented programming that help us simplify the system's complexity by hiding features that are doesn't need to know and only showing what is relevant and making the code more reusable.

Consider the BankAccount Class

```

1  class BankAccount
2  {
3      private string _accountNumber;
4      private float _balannce;
5
6
7      public BankAccount( string AccountNo, Balance)
8      {
9          _accountNumber = AccountNo;
10         _balannce = Balance;
11
12     }
13
14
15     public string AccountNo
16     {
17         get
18         {
19             return _accountNumber;
20         }
21         set
22         {
23             _accountNumber = AccountNo;
24         }
25     }
26     public float Balance
27     {
28         get
29         {
30             return _balannce;
31         }
32         set
33         {
34             _balannce = Balance;
35         }
36     }
37     public void Deposits(float money) {
38         _balannce += money;
39     }
40     public void Withdraw(float money) {
41         if (money > _balannce)
42         {
43             Console.WriteLine("Out of limit");
44         }
45         _balannce -= money;
46     }
47
48 }

```

People wont going to need to know the private fields like the `_accountNumber`, `_ balance` how it is created and how it works, they just need to know their `AccountNo` and the `Balance` with the `Deposit` and the `Withdraw` without knowing what the fields are created and used for. That is the implementation of abstraction that simplifies how people use bank Accounts.

3. What was the issue with the original design in Task 1? Consider what would happen if we had 50 different summary approaches to choose from instead of just 2.

The main issue of the First design is that it is not applied Object Oriented Program structure which makes the code insufficient. Imagine if there are not 2 but 50 new strategies applied in the code, we need to write 50 different strategies that take time and even maybe unreadable to fix the bugs (if exist) or has updates for the code in the future.