1. **Abstraction:** is basically the process of hiding some data and only exposing the essential information or hiding the implementation details and displaying only essential features of an object.
   To demonstrate the principle of abstraction. Classes should have specific responsibilities and collaborate with each other. In the Final Project the following classes video and comment have specific responsibilities and these classes collaborate with each other: The following are the code I used to demonstrate this principle of abstraction:

```
public class Comment
{
    private string _userName;
    private string _comment;

    public Comment(string userName, string comment)
    {
        _userName = userName;
        _comment = comment;

    }
    //GETTERS
    public string GetUserName()
    {
        return _userName;
    }
    public string GetComment()
    {
        return _comment;
    }
}
```

And Also, this:

```
public class Video
{
    private string _author;
    private string _title;
    private int _lenght;
    private List<Comment> _comments;
```

```
    public Video(string author, string title, int lenght)
    {
        _author = author;
        _title = title;
        _lenght = lenght;
        _comments = new List<Comment>();
    }
    public void AddComment(string author, string comment)
    {
        Comment newComment = new Comment(author, comment);
        _comments.Add(newComment);
    }
    public void DisplayVideoInfo()
    {
        Console.WriteLine("\n--- VIDEO INFO ---");
        Console.WriteLine($"Title: {_title}");
        Console.WriteLine($"Author: {_author}");
        Console.WriteLine($"Lenght in seconds: {_lenght}");
        Console.WriteLine($"Lenght in HH:MM:SS: {GetLenghtInHHMMSS()}");
        Console.WriteLine($"Number of comments: {GetCommentsCount()}");
    }
    public void DisplayVideoComments()
    {
        Console.WriteLine("\n--- COMMENTS ---");
        foreach(Comment comment in _comments)
        {
            Console.WriteLine($"{comment.GetUserName()}\n
{comment.GetComment()}");
        }
    }
    public string GetLenghtInHHMMSS()
    {
        TimeSpan time = TimeSpan.FromSeconds(_lenght);
        return time.ToString(@"hh\:mm\:ss");

    }

    //GETTERS
    public string GetAuthor()
    {
```

```
        return _author;
    }
    public string GetTitle()
    {
        return _title;
    }
    public int GetLenght()
    {
        return _lenght;
    }
    public int GetCommentsCount()
    {
        return _comments.Count();
    }
}
```

2. **Encapsulation:** is one the principle of object orientates programming it mean hiding internal state of functionality of object and only allowing access to the public function. To demonstrate the principle of encapsulation. Internal details should be private, external methods should be available for public behaviors. The following code from my program demonstrate this principle:

```
public class Customer

{

    private string _name;

    private Address _addres;



    public Customer(string name, Address address)

    {

        _name = name;

        _addres = address;
```

```
    }
    public string GetName()
    {
        return _name;
    }
    public bool GetIsFromUSA()
    {
        return _addres.GetLivesInUSA();
    }
    public Address GetAddress()
    {
        return _addres;
    }

}
```

3. **Inheritance:** is the programming concept of one or more child classes receiving fields, methods, et cetera from a common parent. It is like people inherit genes from their parents. In programming we can do something similar. Parents classes can have children and anything that parent class has the child class now have. To demonstrate the principle of inheritance. Code should not be duplicated that could be place in a base class. This is code from my program demonstrates the principle:

```
public class LectureEvent : Event
{
    private string _speaker;
    private string _limitedCapacity;
```

```
    public LectureEvent(string title, string description, string date, string time,
Address address, string speaker, string limitedCapacity) : base(title, description,
date, time, address, "Lecture")
    {
        _speaker = speaker;
        _limitedCapacity =  limitedCapacity;
    }
    public override string FullDetails()
    {
        return StandardDetails() + $"\nSpeaker: {_speaker}\nCapacity:
{_limitedCapacity}\n";
    }
}
```

4. Meaning of Polymorphism: Is Greek word that means to "have many forms" objects can be identified by more than one type. Example a Dog also can call: Canine, Animal, Organism. To demonstrate the principle of polymorphism. Methods that are unique to a derived class should be defined in a base class and overridden in a derived class.

```
public class CyclingActivity : Activity
{
    private double _speed;

    public CyclingActivity(string date, double speed, int time) : base(date, "Cycling
Activity", time)
    {
        _speed = speed;
    }
    protected override double GetDistance()
    {
        return (_speed / 60.0) * GetTimeInMinutes();
    }
    protected override double GetSpeed()
    {
        return _speed;
    }
    protected override double GetPace()
    {
```

```
        return GetTimeInMinutes() / GetDistance();
    }
}
```