1. enumeration:
   1. what is enumeration?
      An enum is a user-defined data type that has a fixed set of related values
   2. how to creat enum?
      Use enum to create an enumeration

```
enum Season {
 case spring, summer, autumn, winter
}
```

   3. enum with data type?

```
enum mealSize:Int{
    case small
    case medium
    case large
} let userMealSize:mealSize = .small
print(userMealSize.rawValue)
```

   4. enum associated values?
      attach additional information to an enum case

```
enum Season {
  case name(String)
 case month(Int)
  } var sea=Season.name("summer")
var mon=Season.month(7)
print (sea,mon)
```

2. functions:
   1. how to create function?
      Use func to declare a function. Call a function by following its name with a list of arguments in parentheses.
      Use -> to separate the parameter names and types from the function's return type."

```
func sum(num1: Int,num2 : Int)-> Int{
return num1+num2
}
print(sum(num1: 5,num2:6)) //11
```

2. how to return multiple values?

   Use a tuple

```
 func op(num1: Int,num2 : Int)-> (sum: Int ,
mul: Int){
 var sum = num1+num2
 var mul = num1*num2
 return (sum,mul)
 }  print(op(num1: 5,num2:6)) //11,30
```

3. how to accept variadic parameters?

   append three dots after the parameter's data type.

4. how to assign default parameter value?

   by writing an = after its type followed by the default you want to give it.

5. what is inout parameters?

   inout parameters allow you to change an input passed into a function

```
 func change(_ num: inout Int){
     num = 2
 }  var num = 1
 change(&num)
 print(num)
```

3. Class & struct:

1. how to create

   Use class followed by the class's name to create a class.

```
 class shape{
 var num1=4,num2=5
 func op()-> (sum: Int , mul: Int){
       let sum = num1+num2
       let mul = num1*num2
       return (sum,mul)
 }}
 let c = shape()
 let v = c.op()
    print(v)
```

2. what is the difference between both
   One of the most important differences between structures and classes is that structures are always copied when they're passed around in your code, but classes are passed by reference.
3. what is init and deinit
   Use init to set up the class when an instance is created.
   Use deinit to create a deinitializer if you need to perform some cleanup before the object is deallocated.
4. build models for these entities (Employee, Company, Visa) with available properties and objects