1. Write a function that calculates and returns temperature in Celsius and Kelvin. This function takes input as Fahrenheit which is of type double

func convertTemp(\_ fahren:Double)->(kelvin:Double, celsius:Double){

let kelvin = fahren + 273.15

let celsius = (((fahren \* 5)/9) + 32)

print(kelvin)

print(celsius)

return (kelvin,celsius)

}

1. Compare and contrast between Structure and class with an example.

Struct: Structs are complex data types, they are made up of multiple values. We can create an instance of the struct and fill in its values, then we can pass it around as a single value in your code.

Class: Classes are **building blocks**. Like constants, variables and functions the user can define class properties and methods. Swift 4 provides us the functionality that while declaring classes the users need not create interfaces or implementation files.

**Comparison between Struct and Swift:**

* Define properties to store values
* Define methods to provide functionality
* Define subscripts to provide access to their values using subscript syntax
* Define initializers to set up their initial state
* Be extended to expand their functionality beyond a default implementation
* Conform to protocols to provide standard functionality of a certain kind

**Contrast between Struct and swift:**

Classes have additional capabilities that structures don’t have:

* Inheritance enables one class to inherit the characteristics of another.
* Type casting enables you to check and interpret the type of a class instance at runtime.
* Deinitializers enable an instance of a class to free up any resources it has assigned.
* Reference counting allows more than one reference to a class instance.

1. Write an expected o/p of the following code:

var asianCountries:Set<String> = ["India","China","Nepal",

"Srilanka","Japan","Russia"]

var europeanCountries:Set<String>=["France","Germany",

"Italy","Spain","Russia","Italy"]

var unionAsianEuropean:Set<String> = asianCountries.union(europeanCountries)

print(unionAsianEuropean)

A: ["Russia", "Japan", "Germany", "Spain", "France", "Italy", "India", "China", "Srilanka", "Nepal"]

1. What is the significance of the method prepare(for: sender:)?

A: It is a inbuilt method provided in Swift UIKit. It enables the interactivity with Segue actions in Main storyboard. Whenever a user, creates a segue and it gets called and notifies the view controller that a segue is about to be performed.

1. Write a function name along with the input parameters which is used to get the number of items in the collection view.

A: func collectionView(\_ collectionView: UICollectionView, numberOfItemsInSection section: Int) -> Int {

return movies.count

}

1. Write a function name along with input parameters which returns a cell that corresponds to the specified item in the collection view.

A:

**func** collectionView(\_ collectionView: UICollectionView, cellForItemAt indexPath: IndexPath) -> UICollectionViewCell {

**let** cell = collectionViewOutlet.dequeueReusableCell(withReuseIdentifier: "movie", **for**: indexPath) **as**! MovieCollectionViewCell

cell.assignMovie(with: movies[indexPath.row])

**return** cell

}

1. Consider an image view of width 150 and height 150 and the x , y coordinates of the image are at 50,60 respectively. Write a code such that its width and height expands to 20 points with some animation and falls back to original position after a second.

**@IBAction** **func** buttonClicked(\_ sender: UIButton) {

**var** w = imageOutlet.frame.width

w += 20

**var** h = imageOutlet.frame.height

h += 20

**let** x = imageOutlet.frame.origin.x

**let** y = imageOutlet.frame.origin.y

**let** largerFrame = CGRect(x: x, y: y, width: w, height: h)

UIView.animate(withDuration: 1, delay: 0, usingSpringWithDamping: 0.4, initialSpringVelocity: 50, animations: {

**self**.imageOutlet.frame = largerFrame

})

}