**Que.1 Augmented Reality Template.**

**Code:**

|  |
| --- |
| import UIKit  import SceneKit  import ARKit  class ViewController: UIViewController, ARSCNViewDelegate {  @IBOutlet var sceneView: ARSCNView!    override func viewDidLoad() {  super.viewDidLoad()    sceneView.delegate = self    sceneView.showsStatistics = true    let scene = SCNScene(named: “art.scnassets/ship.scn")!    sceneView.scene = scene  }    override func viewWillAppear(\_ animated: Bool) {  super.viewWillAppear(animated)    let configuration = ARWorldTrackingConfiguration()  sceneView.session.run(configuration)  }    override func viewWillDisappear(\_ animated: Bool) {  super.viewWillDisappear(animated)    sceneView.session.pause()  }      func session(\_ session: ARSession, didFailWithError error: Error) {    }    func sessionWasInterrupted(\_ session: ARSession) {    }    func sessionInterruptionEnded(\_ session: ARSession) {    }  } |

**Que. 2 Single View App**

**Code:**

|  |
| --- |
| import UIKit  import SceneKit  import ARKit  class ViewController: UIViewController, ARSCNViewDelegate {  @IBOutlet var sceneView: ARSCNView!    override func viewDidLoad() {  super.viewDidLoad()    sceneView.delegate = self    sceneView.showsStatistics = true    let scene = SCNScene(named: “art.scnassets/ship.scn")!    sceneView.scene = scene  }    override func viewWillAppear(\_ animated: Bool) {  super.viewWillAppear(animated)    let configuration = ARWorldTrackingConfiguration()  sceneView.session.run(configuration)  } |

**Que. 3 Object Structure**

**Code:**

|  |
| --- |
| import UIKit  import ARKit  import SceneKit  class ViewController: UIViewController, ARSCNViewDelegate {  @IBOutlet var sceneView: ARSCNView!  let config = ARWorldTrackingConfiguration()    func object() {  let shape = SCNBox(width: 0.1, height: 0.1, length: 0.1, chamferRadius: 0.1)  let node = SCNNode()  node.geometry = shape  node.geometry?.firstMaterial?.diffuse.contents = UIColor.orange  node.position = SCNVector3(0,0,-5)  let scene = SCNScene()  sceneView.scene = scene  scene.rootNode.addChildNode(node)  }    @IBAction func resetBtn(\_ sender: UIButton) {  sceneView.session.pause()  sceneView.session.run(config, options: [.resetTracking])  }    override func viewDidLoad() {  super.viewDidLoad()    sceneView.delegate = self  object()  }  } |

**Que. 4 World origin and Feature Points**

**Code:**

|  |
| --- |
| import UIKit  import ARKit  import SceneKit  class ViewController: UIViewController, ARSCNViewDelegate {  @IBOutlet var sceneView: ARSCNView!  let config = ARWorldTrackingConfiguration()    func object() {  let shape = SCNBox(width: 0.1, height: 0.1, length: 0.1, chamferRadius: 0.1)  let node = SCNNode()  node.geometry = shape  node.geometry?.firstMaterial?.diffuse.contents = UIColor.orange  node.position = SCNVector3(0,0,-5)  let scene = SCNScene()  sceneView.scene = scene  scene.rootNode.addChildNode(node)  }    @IBAction func resetBtn(\_ sender: UIButton) {  sceneView.session.pause()  sceneView.session.run(config, options: [.resetTracking])  }    override func viewDidLoad() {  super.viewDidLoad()    sceneView.delegate = self  sceneView.debugOptions = [ARSCNDebugOptions .showWorldOrigin, ARSCNDebugOptions .showFeaturePoints]  object()  }  } |

**Que. 5 Multiple objects**

**Code:**

|  |
| --- |
| import UIKit  import ARKit  import SceneKit  class ViewController: UIViewController, ARSCNViewDelegate {  @IBOutlet var sceneView: ARSCNView!  let config = ARWorldTrackingConfiguration()    @IBAction func boxBtn(\_ sender: UIButton) {  let shape = SCNBox()  let node = SCNNode()  node.geometry = shape  node.geometry?.firstMaterial?.diffuse.contents = UIColor.orange  node.position = SCNVector3(0,0,-5)  let scene = SCNScene()  sceneView.scene = scene  scene.rootNode.addChildNode(node)  }  @IBAction func sphereBtn(\_ sender: UIButton) {  let shape = SCNSphere()  let node = SCNNode()  node.geometry = shape  node.geometry?.firstMaterial?.diffuse.contents = UIColor.orange  node.position = SCNVector3(0,0,-5)  let scene = SCNScene()  sceneView.scene = scene  scene.rootNode.addChildNode(node)  }  @IBAction func coneBtn(\_ sender: UIButton) {  let shape = SCNCone()  let node = SCNNode()  node.geometry = shape  node.geometry?.firstMaterial?.diffuse.contents = UIColor.orange  node.position = SCNVector3(0,0,-5)  let scene = SCNScene()  sceneView.scene = scene  scene.rootNode.addChildNode(node)  }  @IBAction func cylinderBtn(\_ sender: UIButton) {  let shape = SCNCylinder()  let node = SCNNode()  node.geometry = shape  node.geometry?.firstMaterial?.diffuse.contents = UIColor.orange  node.position = SCNVector3(0,0,-5)  let scene = SCNScene()  sceneView.scene = scene  scene.rootNode.addChildNode(node)  }  @IBAction func pyramidBtn(\_ sender: UIButton) {  let shape = SCNPyramid()  let node = SCNNode()  node.geometry = shape  node.geometry?.firstMaterial?.diffuse.contents = UIColor.orange  node.position = SCNVector3(0,0,-5)  let scene = SCNScene()  sceneView.scene = scene  scene.rootNode.addChildNode(node)  }  @IBAction func torusBtn(\_ sender: UIButton) {  let shape = SCNTorus()  let node = SCNNode()  node.geometry = shape  node.geometry?.firstMaterial?.diffuse.contents = UIColor.orange  node.position = SCNVector3(0,0,-5)  let scene = SCNScene()  sceneView.scene = scene  scene.rootNode.addChildNode(node)  }  @IBAction func tubeBtn(\_ sender: UIButton) {  let shape = SCNTube()  let node = SCNNode()  node.geometry = shape  node.geometry?.firstMaterial?.diffuse.contents = UIColor.orange  node.position = SCNVector3(0,0,-5)  let scene = SCNScene()  sceneView.scene = scene  scene.rootNode.addChildNode(node)  }  @IBAction func floorBtn(\_ sender: UIButton) {  let shape = SCNFloor()  let node = SCNNode()  node.geometry = shape  node.geometry?.firstMaterial?.diffuse.contents = UIColor.orange  node.position = SCNVector3(0,0,-5)  let scene = SCNScene()  sceneView.scene = scene  scene.rootNode.addChildNode(node)  }  @IBAction func resetBtn(\_ sender: UIButton) {  sceneView.session.pause()  sceneView.session.run(config, options: [.resetTracking])  }    override func viewDidLoad() {  super.viewDidLoad()    sceneView.delegate = self  sceneView.debugOptions = [ARSCNDebugOptions .showWorldOrigin, ARSCNDebugOptions .showFeaturePoints]    }  } |

**Que. 6 Change the position using slider.**

**Code:**

|  |
| --- |
| import UIKit  import ARKit  import SceneKit  class ViewController: UIViewController, ARSCNViewDelegate {    @IBOutlet var sceneView: ARSCNView!  @IBOutlet var xAxis: UISlider!  @IBOutlet var yAxis: UISlider!  @IBOutlet var zAxis: UISlider!  let configuration = ARWorldTrackingConfiguration()    @IBAction func addObject(\_ sender: UIButton) {  let shape = SCNBox()  let node = SCNNode()  node.geometry = shape  node.geometry?.firstMaterial?.diffuse.contents = UIColor.orange  node.position = SCNVector3(xAxis.value, yAxis.value, zAxis.value)  node.name = "box"  let scene = SCNScene()  sceneView.scene = scene  scene.rootNode.addChildNode(node)  }    @IBAction func resetBtn(\_ sender: UIButton) {  sceneView.session.pause()    sceneView.scene.rootNode.enumerateChildNodes { (node, \_) in  if node.name == "box" {  node.removeFromParentNode()  }  }  sceneView.session.run(configuration, options: [.resetTracking])  }    override func viewDidLoad() {  super.viewDidLoad()    sceneView.delegate = self  sceneView.showsStatistics = true  sceneView.debugOptions = [ARSCNDebugOptions .showWorldOrigin, ARSCNDebugOptions .showFeaturePoints]    }    override func viewWillAppear(\_ animated: Bool) {  super.viewWillAppear(animated)  sceneView.session.run(configuration)  }  } |

**Que. 7 Add texture on object**

**Code:**

|  |
| --- |
| import UIKit  import ARKit  import SceneKit  class ViewController: UIViewController, ARSCNViewDelegate {  @IBOutlet var sceneView: ARSCNView!  let config = ARWorldTrackingConfiguration()    func object() {  let shape = SCNSphere(radius: 0.1)  let node = SCNNode()  node.geometry = shape  node.geometry?.firstMaterial?.diffuse.contents = UIImage(named: “texture.png”)  node.position = SCNVector3(0,0,-5)  let scene = SCNScene()  sceneView.scene = scene  scene.rootNode.addChildNode(node)  }    override func viewDidLoad() {  super.viewDidLoad()    sceneView.delegate = self  object()  }  } |