---

- **Quiz 5: Optionals, Type casting, and Inspection**
- **Closed**
- **Due** Oct 13 at 1:35pm
- Oct 13 at 1:35pm
- 9 pts
- 3 Questions
  - Quiz
- **Quiz 6: Guards and Enumeration**
- **Closed**
- **Due** Oct 20 at 1:35pm
- Oct 20 at 1:35pm
- 6 pts
- 2 Questions
  - Quiz
- **Quiz 6: Protocols and Saving data**
- **Closed**
- **Due** Oct 27 at 1:35pm
- Oct 27 at 1:35pm
- 9 pts
- 2 Questions
  - Quiz
- **Quiz 7: Closures**
- **Closed**
- **Due** Nov 3 at 1:35pm
- Nov 3 at 1:35pm
- 5 pts
- 2 Questions
  - Quiz
- **Exam 2**
- **Not available until** Nov 5 at 2:20pm
- Nov 5 at 2:20pm
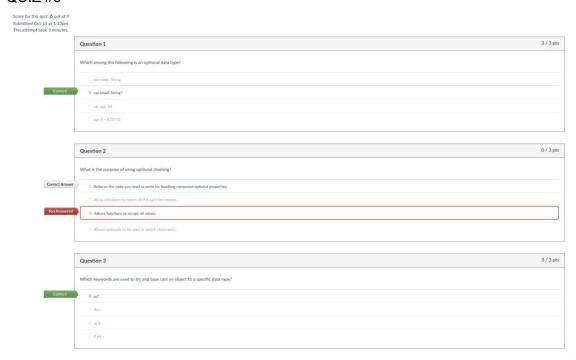- **Due** Nov 5 at 3:20pm
- Nov 5 at 3:20pm
- 50 pts
- 14 Questions

## QUIZ #5

Score for this quiz: **6** out of 9
Submitted Oct 13 at 1:33pm
This attempt took 3 minutes.

### Question 1 — 3 / 3 pts

Which among the following is an optional data type?

- ○ var name: String
- ● var email: String?  **Correct!**
- ○ var age: Int
- ○ var id = 872112

### Question 2 — 0 / 3 pts

What is the purpose of using optional chaining?

- ○ Reduces the code you need to write for handling composed optional properties.  **Correct Answer**
- ○ Allow initializers to return nil if it can't be created.
- ● Allows functions to accept nil values.  **You Answered**
- ○ Allows optionals to be used in switch statements.

### Question 3 — 3 / 3 pts

Which keywords are used to try and type cast an object to a specific data type?

- ● as?  **Correct!**
- ○ Any
- ○ as if
- ○ if let

Quiz Score: **6** out of 9

QUIZ #6:

### Question 1    3 / 3 pts

Complete the guard statement so that the failable initializer returns nil if the end point is before the starting point of the running route. Otherwise, the initializer should create the RunRoute.

```
struct RunRoute {
 var start: Int
 var end: Int
 init?(start: Int, end: Int) {
   guard start < end  else {
    return nil
   }
   self.start = start
   self.end = end
 }
}
```

Answer 1:

Correct!    start < end

### Question 2    3 / 3 pts

Assuming that the eat function accepts a Fruit enumeration value as a parameter, how would you correctly call it?

```
enum Fruit {
 case apple, banana, avocado
}
eat(fruit: .apple )
```
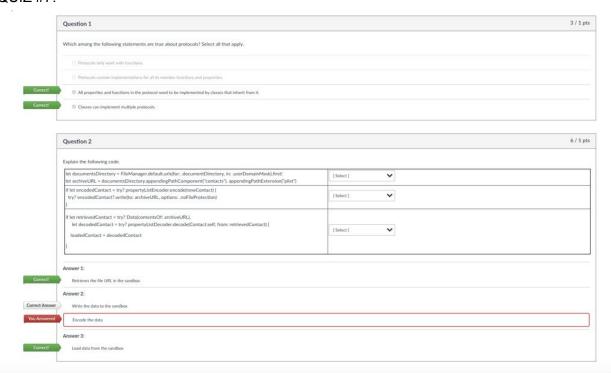
Answer 1:

Correct!    .apple

## QUIZ #7:

---

**Question 1**                                                                3 / 1 pts

Which among the following statements are true about protocols? Select all that apply.

☐ Protocols only work with functions.

☐ Protocols contain implementations for all its member functions and properties.

**Correct!**    ☑ All properties and functions in the protocol need to be implemented by classes that inherit from it.

**Correct!**    ☑ Classes can implement multiple protocols.

---

**Question 2**                                                                6 / 1 pts

Explain the following code.

| | |
|---|---|
| let documentsDirectory = FileManager.default.urls(for: .documentDirectory, in: .userDomainMask).first!<br>let archiveURL = documentsDirectory.appendingPathComponent("contacts"). appendingPathExtension("plist") | [ Select ]  ▼ |
| if let encodedContact = try? propertyListEncoder.encode(newContact) {<br>    try? encodedContact?.write(to: archiveURL, options: .noFileProtection)<br>} | [ Select ]  ▼ |
| if let retrievedContact = try? Data(contentsOf: archiveURL),<br>    let decodedContact = try? propertyListDecoder.decode(Contact.self, from: retrievedContact) {<br>    loadedContact = decodedContact<br>} | [ Select ]  ▼ |

Answer 1:

**Correct!**    Retrieves the file URL in the sandbox

Answer 2:

**Correct Answer**    Write the data to the sandbox

**You Answered**    Encode the data

Answer 3:

**Correct!**    Load data from the sandbox

Quiz S

QUIZ #8:

---

**Question 1**

Which among the following code snippets show the use of closures? Select all that apply.

**Correct!**

```swift
func getEncoder(_ encoder: String) -> (String) -> String {
  if encoder == "Reverse" {
    return {
      value -> String in
      return String(value.reversed())
    }

  } else if encoder == "Inserter" {
    return {
      value -> String in
      var newValue = ""
      for symbol in value {
        newValue.append(symbol)
        newValue.append("aeiou".randomElement()!)
      }
      return newValue
    }
  }
  return {
    value -> String in
    return String(value.reversed())
  }
}
```

**Correct!**

```swift
var grades = [98.0, 97.0, 88.0, 95.0]

let sum: Double = grades.reduce(0.0) {
  (runningValue, newValue) -> Double in
  return runningValue + newValue
}
print("Average: \(sum/Double(grades.count))")
```

**Correct!**

```swift
struct CSUFClass {
  var students: [String]
  var instructor: String
  var location: String
  var schedule: String
  var saveStrategy: (CSUFClass, String) -> Void

  func save(filename: String) {
    saveStrategy(self, filename)
  }
}
```

**Correct!**

```swift
func display(numericalData: [Int], view: (Int)->Void) {
  for num in numericalData {
    view(num)
  }
}
```

**Brian's Study Guide notes**
**CPSC-223W Swift**
**Midterm #2**
**ALL QUIZZES**
**Thursday, November 5, 2020**

---

**Question 2**                                                                                      2 / 2 pts

What is a closure?

Correct!    ● A block of code that can be passed as values.

○ Defines a blueprint for properties and methods that need to be implemented by whatever construct implements it.

○ A special function that calls itself.

○ A special type of property that gets and sets values from other properties.