『iPhone のゲームアプリをつくろう！』
＜正誤表＞

●51 ページ 「手順」のプログラム

【誤】

```
override func touchesBegan(touches: NSSet, withEvent event: UIEvent) {
    for touch: AnyObject in touches {
        let location = touch.locationInNode(self)
        let touchNode = self.nodeAtPoint(location)
        if touchNode.name == "btnLabel" {
            let omikuji = ["大吉","中吉","吉","凶"]
            let r = Int(arc4random_uniform(4))
            let myLabel = self.childNodeWithName("myLabel") as SKLabelNode
            myLabel.text = omikuji[r]
        }
    }
}
```

※5 行目の"btnLabel"のダブルクォーテーション"は不要です。
※薄い青で反転している 4 行のうち 3 行目は不要です。
※薄い青で反転している行の冒頭の let は var になります。
※修正後のプログラムは以下の通りです。

【正】

```
override func touchesBegan(touches: Set<NSObject>, withEvent event: UIEvent) {
    for touch: AnyObject in touches {
        let location = touch.locationInNode(self)
        let touchNode = self.nodeAtPoint(location)
        if touchNode == btnSprite {
            var omikuji = ["大吉","中吉","吉","凶"]
            var r = Int(arc4random_uniform(4))
            myLabel.text = omikuji[r]
        }
    }
}
```

●51 ページ　入力するプログラム 9

【誤】

```
let omikuji = ["大吉","中吉","吉","凶"]
let r = Int(arc4random_uniform(4))
myLabel.text = omikuji[r]
```

【正】

```
var omikuji = ["大吉","中吉","吉","凶"]
var r = Int(arc4random_uniform(4))
myLabel.text = omikuji[r]
```

●151 ページ　下から 9 行目

【誤】これを 2 進数で「0001」「0010」「0100」「1000」と区別しましょう。この値は、「1 << 1」「1 << 2」「1 << 3」「1 << 4」と表します。

【正】これを 2 進数で「00010」「00100」「01000」「10000」と区別しましょう。この値は、「1 << 1」「1 << 2」「1 << 3」「1 << 4」と表します。

●151 ページ　下段のプログラム

【誤】

```
let category_player:UInt32    = 1 << 1    // 0001
let category_marsh:UInt32     = 1 << 2    // 0010
let category_ground:UInt32    = 1 << 3    // 0100
let category_other:UInt32     = 1 << 4    // 1000
```

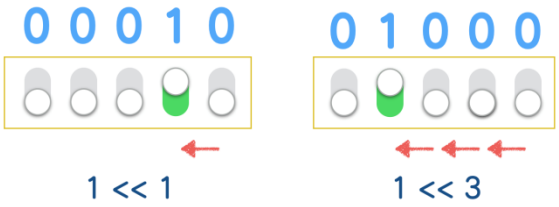
【正】

```
let category_player:UInt32    = 1 << 1    // 00010
let category_marsh:UInt32     = 1 << 2    // 00100
let category_ground:UInt32    = 1 << 3    // 01000
let category_other:UInt32     = 1 << 4    // 10000
```

●152 ページ 「ビットシフト演算子」の説明について
【誤】「0001 を左に 1 回（1回はそのまま）移動させる」と 0001 になり、「0001 を左に 3 回移動させる」と、0100 になります。
【正】「00001 を左に 1 回移動させる」と 00010 になり、「00001 を左に 3 回移動させる」と、01000 になります。

※P151～152 の訂正に伴い、以下のページの図およびプログラムを次のように修正。

＜152 ページ：ビットシフトイラスト＞



＜152 ページ：プログラムキャプチャー画像：let 分のコメント部（赤枠の個所）＞

```
import SpriteKit

class GameScene: SKScene, SKPhysicsContactDelegate {
    // 衝突する物体の種類を用意する
    let category_player:UInt32  = 1 << 1   // 00010
    let category_marsh:UInt32   = 1 << 2   // 00100
    let category_ground:UInt32  = 1 << 3   // 01000
    let category_other:UInt32   = 1 << 4   // 10000
    override func didMoveToView(view: SKView) {
        // 背景色をつける
        self.backgroundColor = UIColor(red: 0.8, green: 0.96, blue: 1, alpha: 1)
        // 物理空間の外枠を作る
        self.physicsBody = SKPhysicsBody(edgeLoopFromRect: self.frame)
        // 物理衝突の情報を自分で受け取る
        self.physicsWorld.contactDelegate = self
        // 物理空間の外枠の種類は、その他
        self.physicsBody?.categoryBitMask = category_other
    }
```

＜153 ページ：プログラムキャプチャー画像：let 分のコメント部（赤枠の個所）＞

```
import SpriteKit

class GameScene: SKScene, SKPhysicsContactDelegate {
    // 衝突する物体の種類を用意する
    let category_player:UInt32  = 1 << 1   // 00010
    let category_marsh:UInt32   = 1 << 2   // 00100
    let category_ground:UInt32  = 1 << 3   // 01000
    let category_other:UInt32   = 1 << 4   // 10000
    // タイマーを用意する
    var myTimer = NSTimer()

    override func didMoveToView(view: SKView) {
        // 背景色をつける
        self.backgroundColor = UIColor(red: 0.8, green: 0.96, blue: 1, alpha: 1)
        // 物理空間の外枠を作る
        self.physicsBody = SKPhysicsBody(edgeLoopFromRect: self.frame)
        // 物理衝突の情報を自分で受け取る
        self.physicsWorld.contactDelegate = self
        // 物理空間の外枠の種類は、その他
        self.physicsBody?.categoryBitMask = category_other
        // タイマーをスタートする（1.0秒ごとにtimerUpdateを繰り返し実行）
        myTimer = NSTimer.scheduledTimerWithTimeInterval(1.0,
            target: self,
            selector: "timerUpdate",
            userInfo: nil,
            repeats: true)
    }
```

＜157 ページ：プログラムキャプチャー画像：let 分のコメント部（赤枠の個所）＞

```
class GameScene: SKScene, SKPhysicsContactDelegate {
    // 衝突する物体の種類を用意する
    let category_player:UInt32  = 1 << 1   // 00010
    let category_marsh:UInt32   = 1 << 2   // 00100
    let category_ground:UInt32  = 1 << 3   // 01000
    let category_other:UInt32   = 1 << 4   // 10000
    // 地面を用意する
    let groundSprite = SKSpriteNode(imageNamed: "ground.png")
    // 障害物の準備をする
    var pinCount = 5
    var pinVX:[CGFloat] = []
    var pinSprite:[SKSpriteNode] = []
    // タイマーを用意する
    var myTimer = NSTimer()
```

●154 ページ　最下段プログラム　最終行
【誤】 self.addChild(popcorn)
【正】 self.addChild(marsh)

●158 ページ　下段プログラム　下から 4 行目
【誤】 pinVX.append(CGFloat(arc4random() % 20) - 10)
【正】 pinVX.append(CGFloat(arc4random_uniform(21)) - 10)

●159 ページ　入力するプログラム 8　下から 2 行目
【誤】 pinVX.append(CGFloat(arc4random() % 20) - 10)
【正】 pinVX.append(CGFloat(arc4random_uniform(21)) - 10)

＜本書サポートサイト＞
http://www.shuwasystem.co.jp/support/7980html/4349.html

＜秀和システム＞
http://www.shuwasystem.co.jp/