```
#include <Servo.h>

Servo tiltServo;
Servo panServo;

const int ledPin          = 8;
const int panServoPin     = 10;
const int tiltServoPin    = 11;
const int horzJoyStickPin = A0;
const int vertJoyStickPin = A1;

int horzJoyStickValue = 0;
int vertJoyStickValue = 0;

int panServoPosition  = 0;
int tiltServoPosition = 0;

const int JSbuttonPin =  9;                    // use pin 9 for the "joystick" button
bool doJSState = true;                         // if doJSState is true, do manual mode.  if
doJSState is false, do program mode
bool newJSButtonState = HIGH;
bool oldJSButtonState = HIGH;

const int LYButtonPin = 2;                     // use pin 2 for the "Left Yes" button
bool doLYState = false;                        // if doLYState is true, do "Left Yes"
bool newLYButtonState = HIGH;
bool oldLYButtonState = HIGH;

const int CYButtonPin = 3;                     // use pin 3 for the "Center Yes" button
bool doCYState = false;                        // if doCYState is true, do "Center Yes"
bool newCYButtonState = HIGH;
bool oldCYButtonState = HIGH;

const int RYButtonPin = 4;                     // use pin 4 for the "Right Yes" button
bool doRYState = false;                        // if doRYState is true, do "Right Yes"
bool newRYButtonState = HIGH;
bool oldRYButtonState = HIGH;

const int LNButtonPin = 5;                     // use pin 5 for the "Left No" button
bool doLNState = false;                        // if doLNState is true, do "Left No"
bool newLNButtonState = HIGH;
bool oldLNButtonState = HIGH;

const int CNButtonPin = 6;                     // use pin 6 for the "Center No" button
bool doCNState = false;                        // if doCNState is true, do "Center No"
bool newCNButtonState = HIGH;
bool oldCNButtonState = HIGH;

const int RNButtonPin = 7;                     // use pin 7 for the "Right No" button
bool doRNState = false;                        // if doRNState is true, do "Right No"
bool newRNButtonState = HIGH;
bool oldRNButtonState = HIGH;
```

```
void setup()
  {
  pinMode(ledPin,      OUTPUT);

  pinMode(JSbuttonPin, INPUT_PULLUP);

  pinMode(LYButtonPin, INPUT_PULLUP);
  pinMode(RYButtonPin, INPUT_PULLUP);
  pinMode(CYButtonPin, INPUT_PULLUP);

  pinMode(LNButtonPin, INPUT_PULLUP);
  pinMode(RNButtonPin, INPUT_PULLUP);
  pinMode(CNButtonPin, INPUT_PULLUP);

  tiltServo.attach(tiltServoPin);
  panServo.attach(panServoPin);

  tiltServo.write(0);
  panServo.write(0);

  randomSeed(analogRead(A6));
  }


void loop()
  {
  if (doJSState)                                                    // if doJSState is
  true
    {
    digitalWrite(ledPin, HIGH);                                     // turn on the
    ledPin   (apply 5 volts to the ledPin)

    horzJoyStickValue = analogRead(horzJoyStickPin);
    panServoPosition  = map(horzJoyStickValue, 0, 1023, 0, 180);
    panServo.write(panServoPosition);

    vertJoyStickValue = analogRead(vertJoyStickPin);
    tiltServoPosition = map(vertJoyStickValue, 0, 1023, 150, 0);
    tiltServo.write(tiltServoPosition);

    bDelay(2);
    }
  else                                                              // otherwise
    {
    digitalWrite(ledPin, LOW);                                      // turn off the ledPin
    (apply 0 volts to the ledPin)

    int progNumber = random(1, 10);

    if (doLYState)
      {
```

```
    progNumber = 6;
    doLYState  = false;
    }
  else if (doRYState)
    {
    progNumber = 7;
    doRYState  = false;
    }
  else if (doCYState)
    {
    progNumber = 2;
    doCYState  = false;
    }
  else if (doLNState)
    {
    progNumber = 8;
    doLNState  = false;
    }
  else if (doRNState)
    {
    progNumber = 9;
    doRNState  = false;
    }
  else if (doCNState)
    {
    progNumber = 3;
    doCNState  = false;
    }

  switch(progNumber)
    {
    case 1:
      doRandom();
      break;

    case 2:
      doYes(5, 90);
      break;

    case 3:
      doNo(5, 80, 90);
      break;

    case 4:
      doScan(3);
      break;

    case 5:
      doRandom();
      doYes(random(3, 15),random(0, 181));
      doNo(random(3, 15), random(0, 151), random(0, 181));
      doScan(3);
      break;
```

```
        case 6:
          doYes(5, 180);
          break;

        case 7:
          doYes(5, 0);
//        doYes(5, 45);

          break;

        case 8:
          doNo(5, 80, 180);
          break;

        case 9:
          doNo(5, 80, 0);
          break;

        default:
           break;
        }

      doJSState = true;               // set to manual mode
      }
    }


void doRandom()
  {
  int howManyTimes = random(3 ,15);

  for (int i = 0; i < howManyTimes; i++)
    {
    tiltServo.write(random(0, 150));
    panServo.write(random(0, 180));
    bDelay(80);
    if (doJSState || doLYState || doCYState || doRYState || doLNState || doCNState ||
    doRNState) {return;}
    }
  }


void doYes(int noOfTimes, int panPosition)
  {
  panPosition = constrain(panPosition, 0, 180);

  panServo.write(panPosition);

  for (int i = 0; i < noOfTimes; i++)
    {
    tiltServo.write(50);
    bDelay(20);
```

```
    if (doJSState || doLYState || doCYState || doRYState || doLNState || doCNState ||
    doRNState) {return;}

    tiltServo.write(125);
    bDelay(20);
    if (doJSState || doLYState || doCYState || doRYState || doLNState || doCNState ||
    doRNState) {return;}
    }
  }


void doNo(int noOfTimes, int tiltPosition, int panPosition)
  {
  panPosition  = constrain(panPosition,  35, 145);
  tiltPosition = constrain(tiltPosition, 50, 120);

  tiltServo.write(tiltPosition);

  for (int i = 0; i < noOfTimes; i++)
    {
    if ((panPosition < 90))
      {
      panServo.write(panPosition - 35);
      bDelay(20);
      if (doJSState || doLYState || doCYState || doRYState || doLNState || doCNState ||
      doRNState) {return;}

      panServo.write(panPosition + 35);
      bDelay(20);
      if (doJSState || doLYState || doCYState || doRYState || doLNState || doCNState ||
      doRNState) {return;}
      }
    else
      {
      panServo.write(panPosition + 35);
      bDelay(20);
      if (doJSState || doLYState || doCYState || doRYState || doLNState || doCNState ||
      doRNState) {return;}

      panServo.write(panPosition - 35);
      bDelay(20);
      if (doJSState || doLYState || doCYState || doRYState || doLNState || doCNState ||
      doRNState) {return;}
      }
    }
  }


void doScan(int noOfTimes)
  {
  tiltServo.write(70);

  for (int i = 0; i <= noOfTimes; i++)
```

```
    {
    for (int j = 0; j < 181; j++)
      {
      panServo.write(j);
      bDelay(1);
      if (doJSState || doLYState || doCYState || doRYState || doLNState || doCNState ||
      doRNState) {return;}
      }

    if (i == 0) {tiltServo.write( 80);}
    if (i == 1) {tiltServo.write(100);}
    if (i == 2) {tiltServo.write(120);}
    if (i == 3) {tiltServo.write(140);}

    for (int j = 180; j >= 0; j--)
      {
      panServo.write(j);
      bDelay(1);
      if (doJSState || doLYState || doCYState || doRYState || doLNState || doCNState ||
      doRNState) {return;}
      }

    if (i == 0) {tiltServo.write( 90);}
    if (i == 1) {tiltServo.write(110);}
    if (i == 2) {tiltServo.write(130);}
    if (i == 3) {tiltServo.write(150);}
    }
  }


void bDelay(int noOf10Milliseconds)
  {
  for (int i = 0; i < noOf10Milliseconds; i++)
    {
    delay(10);

    buttonCheck(newLYButtonState, oldLYButtonState, doLYState, LYButtonPin);
    if (doLYState) {doJSState = false; return;}                                    // if this
    button was pressed, set to program mode and return

    buttonCheck(newRYButtonState, oldRYButtonState, doRYState, RYButtonPin);
    if (doRYState) {doJSState = false; return;}                                    // if this
    button was pressed, set to program mode and return

    buttonCheck(newCYButtonState, oldCYButtonState, doCYState, CYButtonPin);
    if (doCYState) {doJSState = false; return;}                                    // if this
    button was pressed, set to program mode and return

    buttonCheck(newLNButtonState, oldLNButtonState, doLNState, LNButtonPin);
    if (doLNState) {doJSState = false; return;}                                    // if this
    button was pressed, set to program mode and return

    buttonCheck(newRNButtonState, oldRNButtonState, doRNState, RNButtonPin);
```

```
    if (doRNState) {doJSState = false; return;}                                    // if this
    button was pressed, set to program mode and return

    buttonCheck(newCNButtonState, oldCNButtonState, doCNState, CNButtonPin);
    if (doCNState) {doJSState = false; return;}                                    // if this
    button was pressed, set to program mode and return

    buttonCheck(newJSButtonState, oldJSButtonState, doJSState, JSbuttonPin);
    if (doJSState) {return;}                                                       // if
    joystick button was pressed, return
    }
  }


void buttonCheck(bool &newButtonState, bool &oldButtonState, bool &doState, int buttonPin)
  {
  newButtonState = digitalRead(buttonPin);                              // read the input value of
  the button pin and store it in newButtonState

  if ((newButtonState == LOW) && (oldButtonState == HIGH))              // if the button State goes
  from LOW to HIGH
    {                                                                   // the button has been
    pressed
    doState = !doState;                                                     // change the state to
    "true"
    delay(10);                                                          // wait 10 millisecconds
    (1/100 of a second)
    }

   if((newButtonState == HIGH) && (oldButtonState == LOW))              // if the button State goes
   from HIGH to LOW
    {                                                                   // the button has been
    released
    delay(10);                                                          // wait 10 millisecconds
    (1/100 of a second)
    }

   oldButtonState = newButtonState;                                     // update the value of the
   oldButtonState
  }
```