//set the switches to match with their pair

int switchPin1 = 3;
int switchPin2 = 4;
int switchPin3 = 5;
int switchPin4 = 6;

int ledPin1 = 9;
int ledPin2 = 10;
int ledPin3 = 11;
int ledPin4 = 12;

//assigns booleans
boolean lastButton1 = LOW;
boolean lastButton2 = LOW;
boolean lastButton3 = LOW;
boolean lastButton4 = LOW;

boolean currentButton1 = LOW;
boolean currentButton2 = LOW;
boolean currentButton3 = LOW;
boolean currentButton4 = LOW;

boolean ledOn1 = LOW;
boolean ledOn2 = LOW;
boolean ledOn3 = LOW;
boolean ledOn4 = LOW;

int button[] = {switchPin1, switchPin2, switchPin3, switchPin4}; //creates arrays, so pairs have common indices
int led[] = {ledPin1, ledPin2, ledPin3, ledPin4};
boolean lastButton[] = {lastButton1, lastButton2, lastButton3, lastButton4};
boolean currentButton[] = {currentButton1, currentButton2, currentButton3, currentButton4};
boolean ledOn[] = {ledOn1, ledOn2, ledOn3, ledOn4};

boolean debounce(int butnum) {
    //accounts for physical bouncing when checking a button input
    boolean current = digitalRead(button[butnum]);
    if (lastButton[butnum] != current) //checks to see if the current state has changed
    {
        delay(5); //if it has, then wait .005 seconds to see if it was random fluctuation or not
        current = digitalRead(button[butnum]);
    }
}
boolean waspressed(int butnum){
    //checks to see if the specified button number was pressed. Returns HIGH or LOW
    //Serial.println("Entered Waspressed");
    //Serial.println(HIGH);
    currentButton[butnum] = debounce(butnum);
    //Serial.println(currentButton[butnum]);
    if (currentButton[butnum] != LOW){ //check if a button is being pressed
        //Serial.println(currentButton[butnum]);
        //Serial.println("Pressed");
        currentButton[butnum] = !currentButton[butnum];
        return HIGH;
    } else{
        return LOW;
    }
}

int createsecret(int len, int *secretPointer){
    //Creates a random pattern of 0-3s along a specified memory path
    int choice = 0;
    for(int x = 0; x<len; x = x+1){ //Iterates through the secretarray
        choice = random(0,4); //chooses a random light
        //Serial.println(choice);
        //digitalWrite(led[pattern[choice]], HIGH); //sends the command to turn on the light
        //delay(500); //waits
        //digitalWrite(led[pattern[x]], LOW); //sends the command to turn it off
        //delay(500); //Creates a flash if the button press was the same
        *secretPointer = choice; //Saves the choice into the array
        *secretPointer++;//Increments the secretPointer
    }
    return 0;
}

int equalarrays(int len, int *pointer1, int *pointer2){
    //Evaluates two arrays of equal length to see if they are identical elementwise
    for(int x = 0; x<len; x = x+1){ //Interates through both arrays
        if(*pointer1 == *pointer2){ //Checks for elementwise equality
            *pointer1++;
            *pointer2++;
        }
    }
    return 0;
}
continue;
}
else{
    return LOW; // If they don't match, then return false
}
}
return HIGH; // If everything matches, return true

int getpattern(int len, int *arrayPointer){
    // Captures the button presses of the user and stores them in the specified memory path
    int current = 0; // Pattern counter
    while (current != len){
        for(int n = 0; n<4; n=n+1){
            // Serial.println(current);
            // Serial.println(n);
            if (waspressed(n) == HIGH){ // If a button was pressed
                // Serial.println("pressed");
                // Serial.println(n);
                digitalWrite(led[n], HIGH); // Turn on the light to signal the pattern
                *arrayPointer = n; // Add it to the pattern in the current position in the pattern
                // Serial.println(pattern[current]);
                current = current+1; // Increment the pattern number
                *arrayPointer++; // Increment the array pointer
                while (waspressed(n) == HIGH){ // Wait until the button is released to continue listening
                    delay(1);
                }
                digitalWrite(led[n], LOW); // Turn off the light when the button is released
                if (current == len){
                    break;
                }
            }
        }
    }
    return 0;
}

int displaypattern(int len, int *patternpointer){
    // Displays on the LEDs a pattern at a specified memory location
    for(int x = 0; x<len; x = x+1){ // Iterates through the pattern
//Serial.println(*patternpointer);
  digitalWrite(led[*patternpointer], HIGH); //sends the command to turn on the light
  delay(1000); //waits
  digitalWrite(led[*patternpointer], LOW); //sends the command to turn it off
  delay(500); //Creates a flash if the button press was the same
  patternpointer++;
}
return 0;
}

int flash(int lednum, int times){
  //Flashes a specified led a specified amount of times
  for(int x = 0; x<times; x = x+1){
    digitalWrite(led[lednum], HIGH); //sends the command to turn on the light
    delay(200); //waits
    digitalWrite(led[lednum], LOW); //sends the command to turn it off
    delay(100); //Creates a flash if the button press was the same
  }
  return 0;
}

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600); //initializes the serial interface to print to the computer
  pinMode(switchPin1, INPUT); //initializes the inputs
  pinMode(switchPin2, INPUT);
  pinMode(switchPin3, INPUT);
  pinMode(switchPin4, INPUT);
  pinMode(ledPin1, OUTPUT); //initializes the outputs
  pinMode(ledPin2, OUTPUT);
  pinMode(ledPin3, OUTPUT);
  pinMode(ledPin4, OUTPUT);
  randomSeed(analogRead(0)); //Resets the random seed
  //Serial.println(analogRead(0));
  int randanalog1 = random(0,6); //Picks a random analong port to seed from
  //Serial.println(randanalog1);
  randomSeed(analogRead(randanalog1)); //Resets the random using randanalog
  //Serial.println(analogRead(randanalog));
  long randomNumber1 = random(2147483647); //Creates a really long random
  //Serial.println(randomNumber1);
  int randanalog2 = random(0,6); //Picks a random analong port to seed from
Serial.println(randanalog2);
randomSeed(analogRead(randanalog2)); //Resets the random using randanalog2
long randomNumber2 = random(2147483647); //Does another long random
Serial.println(randomNumber2);
randomSeed(randomNumber2); //Another random seed

//Setting the length of the pattern to guess
int secretlen = 3;
//Setting up the two arrays and their pointers
int secret[100]; //100 just in case you go for a while
int *secretpointer = &secret[0];
int guess[100]; //100 just in case you go for a while
int *guesspointer = &guess[0];

void loop() {
  // put your main code here, to run repeatedly:
  //Serial.println("Made it to loop");
  //Come up with a secret and displays it
  createsecret(secretlen, secretpointer);
  displaypattern(secretlen, secretpointer);
  Serial.println("pattern is");
  for (int i = 0; i < secretlen; i = i + 1) {
    Serial.println(secret[i]);
  }
  //Asks you to guess the pattern
  getpattern(secretlen, guesspointer);

  //Check to see if your guess is equal to the secret or not
  if (equalarrays(secretlen, guesspointer, secretpointer) == LOW) {
    //if your guess was wrong, resets the secretlen and starts over the loop
    secretlen = 3;
    //Tells you that you were wrong by flashing just the red light thrice
    delay(500);
    flash(1, 3);
    delay(1000);
  }
}
else{
    // If your guess was right, ups secretlen by one and continues the loop
    secretlen++;
    // Tells you that you were right by flashing just the green light thrice
    delay(500);
    flash(2, 3);
    delay(1000);
}
}