```
Module[{repeatsBoole, twoPlayerBoole = False, twoPlayer = 0, twoPlayerSelect = False, player1Score, playing = True,
  colors, colorReplacements, solution, tryGuess, submitGuess,
  sequenceGuesser, sequence, keep, select,
  peg, newGameButton, newGame, helper,
  history, historyIterator = 0, numbering,
  numPegs = 4, numPegsSetter = 4,
  numColors = 6, numColorsSetter = 6,
  numGuesses = 10, numGuessesSetter = 10,
  guessed = Table[Null, {i, 1, 24}],
  attempts = 0,
  titleText = "Mastermind : Can you Deduce the Code?",
  buttonText = "Submit Guess"},

 (*Initalization*)
colors = {Red, Green, Blue, Yellow, White, Black, Cyan, Magenta, Orange, Pink, Purple, Brown};
colorReplacements = Table[i → colors[[i]], {i, 1, 12}];
numbering = Table[Style[i, 20], {i, 1, numGuesses}] ~Join~ Table["X", {i, numGuesses + 1, 24}];
solution = RandomInteger[{1, numColors}, numPegs];
sequence = Table[Mod[i, numColors, 1], {i, 1, numPegs}];
keep = Table[False, {i, 1, numPegs}];
select = 1;

 (*Function*)
Clear[tryGuess, repeatsQ];
tryGuess[guess_, solution_] := Block[{compare, checkGuess, checkSolution, guessColors, sequenceLength, i, j},
  sequenceLength = Length@solution;
  compare = Table[0, {sequenceLength}]; checkGuess = Table[0, {sequenceLength}]; checkSolution = Table[0, {sequenceLength}];
  (*Check for exact matches first*)
  For[i = 1, i <= sequenceLength, i++,
   If[checkGuess[[i]] == 0 && guess[[i]] == solution[[i]],
    compare[[i]] = 2; checkGuess[[i]] = 1; checkSolution[[i]] = 1]];
  (*Check for out of place pegs*)
  For[i = 1, i <= sequenceLength, i++,
   For[j = 1, j <= sequenceLength, j++,
    If[checkGuess[[i]] == 0 && checkSolution[[j]] == 0 && guess[[i]] == solution[[j]],
     compare[[i]] = 1; checkGuess[[i]] = 1; checkSolution[[j]] = 1]]];
  guessColors = guess /. colorReplacements;
  compare = Sort[compare, Greater] /. {2 → Black, 1 → White, 0 → GrayLevel[.5]};
  (*Generate Graphics*)
  ToExpression["Graphics[{EdgeForm[Thick],"<>
    Table[ToString[guessColors[[i]]] <> ",
      Rectangle[{" <> ToString[i - 1] <> ",0}]," <>
      ToString[compare[[i]]] <> ",
      Rectangle[" <> ToString[{sequenceLength + .5 Floor[1/2 (i + 1)], .5 Mod[i, 2]}] <>
     "," <> ToString[{sequenceLength + .5 Floor[1/2 (i + 1)] + .5, .5 Mod[i, 2] + .5}] <> "],",
     {i, 1, sequenceLength}] <> "Text[ ,{0,0}]},ImageSize->{250, 42},Background->White]", TraditionalForm] (*ToExpression*)
 ]; (*Block*)

repeatsQ[list_] := Block[{i, j},
  For[i = 1, i ≤ Length@list, i++,
   For[j = 1, j ≤ Length@list, j++,
    Which[i == j, Continue; Print,
     list[[i]] == list[[j]], Return[True],
     True, Continue]]
  ]; Return[False]];

(*Dynamic Objects*)
peg[i_] := Column[{Checkbox[Dynamic[keep[[i]]]], EventHandler[Dynamic[
Graphics[{sequence[[i]] /. colorReplacements,
Dynamic[If[select == i, EdgeForm[Thickness[.04]], EdgeForm[Thickness[.08]]]],
Rectangle[]}, ImageSize -> 40]],
{"MouseClicked" :> (If[sequence[[i]] == numColors, sequence[[i]] = 1, sequence[[i]] += 1])}]
   }, Alignment -> Center];

sequenceGuesser = Dynamic[
  Panel[Grid[{Table[peg[i], {i, 1, numPegs}],
     {Button[buttonText, submitGuess,
       ImageSize → Full], SpanFromLeft}}]
  ], TrackedSymbols :> {numPegs, numColors, buttonText, sequence}];


history := Dynamic[
  Grid[Thread[{numbering[[1 ;; 8]], guessed[[1 ;; 8]],
     numbering[[9 ;; 16]], guessed[[9 ;; 16]],
     numbering[[17 ;; 24]], guessed[[17 ;; 24]]}],
   ItemSize → {{3, Scaled[.28], 3, Scaled[.28], 3, Scaled[.28]}, 3.5},
   Alignment → {Center, Center}, Frame → All],
  TrackedSymbols :> {guessed, numGuesses, historyIterator}];

helper = Panel[Grid[Partition[
    Flatten@
     Table[{Graphics[{colors[[i]], EdgeForm[Thick], Rectangle[]}, ImageSize → 20], InputField[Null, FieldSize → 1]}, {i, 1, 12}], 8]],
  ImageSize → {250, 110}];

(*In Game Events and Logic*)
newGame := (playing = True;
  numPegs = numPegsSetter;
  numColors = numColorsSetter;
  numGuesses = numGuessesSetter;
  guessed = Table[" ", {i, 1, 25}];
  historyIterator = 0;
  If[twoPlayerBoole == True, titleText = "Who is the True Mastermind?"; numGuessesSetter = 25; numGuesses = 25;
   Which[twoPlayer == 0, twoPlayer = 1; buttonText = "Player 1 Enter Code"; twoPlayerSelect = True,
    twoPlayer == 1, twoPlayer = 2; buttonText = "Player 2 Enter Code"; twoPlayerSelect = True],
   twoPlayer = 0; twoPlayerSelect = False; titleText = "Mastermind : Can you Deduce the Code?"; buttonText = "Submit Guess"];
  numbering = Table[Style[i, 20], {i, 1, numGuesses}] ~Join~ Table["X", {i, numGuesses + 1, 24}];
  attempts = 0;
  sequence = Table[Mod[i, numColors, 1], {i, 1, numPegs}];
  keep = Table[False, {i, 1, numPegs}];
  solution = RandomInteger[{1, numColors}, numPegs];
  If[numPegs > numColors, repeatsBoole = False];
  If[repeatsBoole, While[repeatsQ[solution],
    solution = RandomInteger[{1, numColors}, numPegs]]]);

submitGuess := If[playing == True,
  Which[twoPlayerSelect == True,
   twoPlayerSelect = False; solution = sequence; sequence = Table[Mod[i, numColors, 1], {i, 1, numPegs}];
   Which[twoPlayer == 1, buttonText = "Player 2: Submit Guess", twoPlayer == 2, buttonText = "Player 1: Submit Guess"],

   sequence == solution, playing = False;
   guessed[[Mod[attempts, 24] + 1]] = tryGuess[sequence, solution]; titleText = "Congratulations!";
   buttonText = "Press New Game to Play Again";
   Which[twoPlayer == 1, player1Score = attempts; buttonText = "Press New Game to Continue",
    twoPlayer == 2, twoPlayer = 0;
    Which[player1Score < attempts, titleText = "Player 2 is the Mastermind",
     player1Score > attempts, titleText = "Player 1 is the Mastermind",
     player1Score == attempts, titleText = "Tie Match! Play Again!"]],

   numGuesses == 25, guessed[[Mod[attempts, 24] + 1]] = tryGuess[sequence, solution]; attempts++;
   Which[Mod[attempts, 24] == 0 && attempts > 0, historyIterator++; guessed[[1 ;; 8]] = Null;
    numbering[[1 ;; 8]] = Table[Style[24 * historyIterator + i, 20], {i, 1, 8}],
    Mod[attempts, 24] == 8, guessed[[9 ;; 16]] = Null; numbering[[9 ;; 16]] = Table[Style[24 * historyIterator + i, 20], {i, 9, 16}],
    Mod[attempts, 24] == 16, guessed[[17 ;; 24]] = Null; numbering[[17 ;; 24]] = Table[Style[24 * historyIterator + i, 20], {i, 17, 24}]],

   True, Which[attempts < numGuesses - 1, guessed[[attempts + 1]] = tryGuess[sequence, solution]; attempts++,
    attempts == numGuesses - 1, guessed[[attempts + 1]] = tryGuess[sequence, solution];
        sequence = solution; titleText = "You are not the Mastermind."; buttonText = "Correct Solution"; playing = False]
  ]];

(*Game Board*)
EventHandler[Panel[Deploy@Panel[Grid[{
      {Style[Dynamic[titleText], "Title"], SpanFromLeft},
      {Pane[" "], SpanFromLeft},
      {Panel@Column[{Button["New Game", newGame],
          Labeled[Checkbox[Dynamic[twoPlayerBoole]], "2 Player", Left],
          Pane[" "],
          Pane["Number of Pegs"],
          PopupMenu[Dynamic[numPegsSetter], Table[i, {i, 1, 10}]],
          Pane[" "],
          Pane["Number of Colors"],
          PopupMenu[Dynamic[numColorsSetter], Table[i, {i, 1, 12}]],
          Labeled[Checkbox[Dynamic[repeatsBoole]], "No Repeated Colors", Left],
          Pane[" "],
          Pane["Number of Guesses"],
          PopupMenu[Dynamic[numGuessesSetter], Table[i, {i, 1, 24}] ~Join~ {25 → "∞"}], Center](*End Column*), history},
      {Row[{Panel[" "], Appearance -> "Frameless", ImageSize → 255], sequenceGuesser, Panel["    ", Appearance -> "Frameless"], helper}], SpanFromLeft}}],
   ImageSize → {1100, 580}
  ], Background → Black],
 {{"KeyDown", "t"} :> If[twoPlayerBoole == False, twoPlayerBoole = True, twoPlayerBoole = False],
  {"KeyDown", "r"} :> If[repeatsBoole == False, repeatsBoole = True, repeatsBoole = False],
  {"KeyDown", "p"} :> If[numPegsSetter == 10, numPegsSetter = 1, numPegsSetter++],
  {"KeyDown", "c"} :> If[numColorsSetter == 12, numColorsSetter = 1, numColorsSetter++],
  {"KeyDown", "g"} :> If[numGuessesSetter == 25, numGuessesSetter = 1, numGuessesSetter++],
  {"KeyDown", "k"} :> If[keep[[select]] == False, keep[[select]] = True, keep[[select]] = False],
  "EscapeKeyDown" :> newGame,
  "RightArrowKeyDown" :> If[select == numPegs, select = 1, select += 1],
  "LeftArrowKeyDown" :> If[select == 1, select = numPegs, select -= 1],
  "UpArrowKeyDown" :> If[keep[[select]] == False && playing == True,
    If[sequence[[select]] == numColors, sequence[[select]] = 1, sequence[[select]] += 1]],
  "DownArrowKeyDown" :> If[keep[[select]] == False && playing == True,
    If[sequence[[select]] == 1, sequence[[select]] = numColors, sequence[[select]] -= 1]],
  "ReturnKeyDown" :> submitGuess}]
]
```