```
 1  using System;
 2  using static System.Console;
 3  using System.Collections;
 4  using System.Linq;
 5  using System.Text;
 6  using System.Threading.Tasks;
 7
 8  namespace SkillsUSADistrictsV2
 9  {
10      class Driver
11      {
12          static void Main(string[] args)
13          {
14              //Random Number Generator placed here so we are always using the
                  same one
15              Random num = new Random();
16              int starkBattleWins = 0;
17              int capBattleWins = 0;
18
19              /****[ Begin Main Method ]****/
20              Write("How many fighters per team? 1 - 6 are valid answers: " );
21              int numFighters = Convert.ToInt32(ReadLine());
22
23              Write("How many battles should they fight? 1  - 10 are valid
                  answers: ");
24              int numBattles = Convert.ToInt32(ReadLine());
25              WriteLine();
26
27              for (int i = 0; i < numBattles; i++)
28              {
29                  int winner = BattleGenerator(numFighters, num);
30
31                  if(winner == 0)
32                  {
33                      starkBattleWins += 1;
34                  }
35                  else if (winner == 1)
36                  {
37                      capBattleWins += 1;
38                  }//End if / else if
39              }//End for loop
40
41              CalcWarWinner(starkBattleWins, capBattleWins);
42
43              //To Stop The Flow Of Code
44              ReadKey();
45          }
46
47          /****[ BattleGenerator Method ]***************
```

```
48          * EXPECTS: The number of team members, and a
49          *  random number.
50          * RETURNS: Nothing
51          * TASKS:
52          ******************************************/
53         public static int BattleGenerator(int numOfMembers, Random num)
54         {
55             //These two paraelle arrays are for holding data for later use
56             //Similar to a data base and pulling data from it.
57             string[,] teamStark = new string[,] { {"Black Widow", "10"}, { "War
                   Machine", "20" }, { "Spider Man", "30" },
58                                                   {"Black Panther", "40"},
                       {"Iron Man", "50"}, {"Vision", "60"}};
59
60             string[,] teamCap = new string[,] { {"Hawkeye", "10"}, { "Falcon",
                 "20" }, { "Ant-Man", "30" },
61                                                 {"Winter Soldier", "40"},
                       {"Captain America", "50"}, {"Scarlet Witch", "60"}};
62
63             //Create Team Stark
64             ArrayList stark = CreateTeam(numOfMembers, num, teamStark);
65             int starkPower = CalcTeamPowerRank(stark);
66             //Create Team Cap
67             ArrayList cap = CreateTeam(numOfMembers, num, teamCap);
68             int capPower = CalcTeamPowerRank(cap);
69
70             //Print Team Stark
71             WriteLine("--------[ Team Stark ]--------\n");
72             PrintTeamData(stark);
73
74             //Print Team Cap
75             WriteLine("---------[ Team Cap ]---------\n");
76             PrintTeamData(cap);
77
78             //Get the winner of the battle
79             int winner = CalcBattleWinner(starkPower, capPower);
80
81             return winner;
82         }
83
84         /****[ CreateTeam Method ]********************
85          * EXPECTS: The number of people on the team,
86          *  a random number, and the teams information.
87          * RETURNS: An arraylist of team members that
88          *  are on the team and ready for battle.
89          * TASKS: Create an arraylist to hold all the
90          *  team members. Then create the number of
91          *  team members specified. Do this by getting
92          *  a random number, using that number to access
```

```
93          *  out data in the paraelle array. Create a
94          *  team member using the constructor. Call the
95          *  'AddTeamMember' method to validate the team
96          *  member. Return the list of team members for
97          *  the battle.
98          *********************************************/
99          public static ArrayList CreateTeam(int numOfMembers, Random num, string
              [,] teamInfo)
100         {
101             //Declare an array list
102             ArrayList teamMembers = new ArrayList();
103
104             for (int i = 0; i < numOfMembers; i++)
105             {
106                 bool memberAdded = false;
107
108                 while (memberAdded == false)
109                 {
110                     //Generate a number
111                     int arrayNum = num.Next(0, 6);
112
113                     //Create a team member
114                     TeamMember member = new TeamMember(teamInfo[arrayNum, 0],
                        teamInfo[arrayNum, 1]);
115
116                     //Call the add member method of the team class
117                     memberAdded = AddTeamMember(member, teamMembers);
118                 }//End While
119             }//End for loop
120
121             return teamMembers;
122         }//End Create Team
123
124         /****[ AddTeamMember Method ]*****************
125          * EXPECTS: A new team member to add to the
126          *  team and the list of current team members.
127          * RETURNS: A bool true - Member was added to
128          *  the team. bool false - Member is already on
129          *  the team and we need to try again.
130          * TASKS: Check if the new team member is on
131          *  the team, if the new team member is return
132          *  false and if not return true and also add
133          *  the new team member to the list of team
134          *  members.
135          *********************************************/
136         public static bool AddTeamMember(TeamMember memberToAdd, ArrayList
              teamMembers)
137         {
138             bool memberExists = false;
```

```
139
140             for (int i = 0; i < teamMembers.Count; i++)
141             {
142                 TeamMember currentTeamMember = (TeamMember) teamMembers[i];
143                 if (memberToAdd.Name == currentTeamMember.Name)
144                 {
145                     memberExists = true;
146                 }//End if
147             }//End for loop
148
149             if (memberExists == true)
150             {
151                 //false because the team member is on the team so we need to
                      try again
152                 return false;
153             }
154             else
155             {
156                 teamMembers.Add(memberToAdd);
157                 //true because the member was added to the team
158                 return true;
159             }//End if / else
160         }//End AddTeamMember
161
162         /****[ PrintTeamData Method ]*****************
163          * EXPECTS: An arraylist of team members
164          * RETURNS: Nothing
165          * TASKS: Loops through the list of team
166          *  members and prints there name and power rank
167          ******************************************/
168         public static void PrintTeamData(ArrayList teamMemberList)
169         {
170             for (int i = 0; i < teamMemberList.Count; i++)
171             {
172                 TeamMember member = (TeamMember)teamMemberList[i];
173                 Write($"| {member.Name} {member.PowerRank} |");
174             }
175             WriteLine("\n");
176         }//End PrintTeamData
177
178         /****[ CalcTeamPowerRank Method ]*************
179          * EXPECTS: The list of team members
180          * RETURNS: The total power rank of the team
181          * TASKS: Loop through the entire team adding
182          *  up the total power rank for the team and
183          *  then return the total score.
184          ******************************************/
185         public static int CalcTeamPowerRank(ArrayList team)
186         {
```

```
187             int totalScore = 0;
188
189             for (int i = 0; i < team.Count; i++)
190             {
191                 TeamMember member = (TeamMember)team[i];
192                 //Gets each members power rank
193                 totalScore += Convert.ToInt32(member.PowerRank);
194             }
195             return totalScore;
196         }//End CalcTeamPowerRank
197
198         /****[ CalcBattleWinner Method ]**************
199          * EXPECTS: Starks total power rank and caps
200          *  total power rank.
201          * RETURNS: A int where -1 is a tie, 0 stark
202          *  wins, and 1 cap wins.
203          * TASKS: Calculate who won the battle by
204          *  comparing the two teams power ranks.
205          ******************************************/
206         public static int CalcBattleWinner(int starkPower, int capPower)
207         {
208             if(starkPower == capPower)
209             {
210                 WriteLine("*********[ It's A Tie ]*********");
211                 WriteLine($"Team Stark: {starkPower.ToString()} Vs Team Cap:
                      {capPower.ToString()}");
212                 WriteLine("*********************************\n");
213                 return -1;
214             }
215             else if(starkPower > capPower)
216             {
217                 WriteLine("****[ Team Stark Wins ]*********");
218                 WriteLine($"Team Stark: {starkPower.ToString()} Vs Team Cap:
                      {capPower.ToString()}");
219                 WriteLine("*********************************\n");
220                 return 0;
221             }
222             else
223             {
224                 WriteLine("******[ Team Cap Wins ]*********");
225                 WriteLine($"Team Stark: {starkPower.ToString()} Vs Team Cap:
                      {capPower.ToString()}");
226                 WriteLine("*********************************\n");
227                 return 1;
228             }//End if / else if / else
229         }//End CalcBattleWinner
230
231         public static void CalcWarWinner(int starkWins, int capWins)
232         {
```

```
233             if (starkWins == capWins)
234             {
235                 WriteLine("******[ The War Is A Tie ]******");
236                 WriteLine($"Team Stark: {starkWins.ToString()} Vs Team Cap:
                      {capWins.ToString()}");
237                 WriteLine("*********************************\n");
238             }
239             else if (starkWins > capWins)
240             {
241                 WriteLine("**[ Team Stark Wins The War ]***");
242                 WriteLine($"Team Stark: {starkWins.ToString()} Vs Team Cap:
                      {capWins.ToString()}");
243                 WriteLine("*********************************\n");
244             }
245             else
246             {
247                 WriteLine("***[ Team Cap Wins The War ]****");
248                 WriteLine($"Team Stark: {starkWins.ToString()} Vs Team Cap:
                      {capWins.ToString()}");
249                 WriteLine("*********************************\n");
250             }//End if / else if / else
251         }
252     }//End Class
253 }//End Namespace
254
```