```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6  using DragonBallGame.CharacterClass;
 7  using DragonBallGame.TransformationState;
 8
 9  namespace DragonBallGame.CharacterClass
10  {
11      public abstract class Character
12      {
13          private string _name;
14          private int _power;
15          private int _health;
16          private int _maxHealth;
17          private string _specialAbility;
18          private int _transformationLvl;
19          private int _maxLevel;
20          private string _form;
21          private int _energy;
22          private int _maxEnergy = 100;
23          private ITransformationState _transformationState;
24          private bool _isBlocking;
25
26          public Character(string name, int power, int health, string
              specialAbility)
27          {
28              Name = name;
29              Power = power;
30              Health = health;
31              _maxHealth = health;
32              SpecialAbility = specialAbility;
33              TransformationLevel = 0;
34              _form = "Base Form";
35              _energy = 0;
36              _transformationState = new SuperSaiyan1();
37              _isBlocking = false; // Initialize as not blocking
38          }
39
40          public string Name { get => _name; set => _name = value; }
41          public int Power { get => _power; set => _power = value; }
42          public int Health { get => _health; set => _health = value; }
43          public int MaxHealth { get => _maxHealth; set => _maxHealth =
              value; }
44          public string SpecialAbility { get => _specialAbility; set =>
              _specialAbility = value; }
45          public int TransformationLevel { get => _transformationLvl; set =>
               _transformationLvl = value; }
```

```
46          public abstract int MaxLevel { get; }
47          public string Form { get => _form; set => _form = value; }
48
49          public int Energy
50          {
51              get => _energy;
52              set
53              {
54                  if (value > _maxEnergy)
55                      _energy = _maxEnergy;
56                  else if (value < 0)
57                      _energy = 0;
58                  else
59                      _energy = value;
60              }
61          }
62
63          public bool IsBlocking
64          {
65              get { return _isBlocking; }
66          }
67
68          // Method to start blocking
69          public void Block()
70          {
71              _isBlocking = true;
72          }
73
74          // Method to stop blocking
75          public void StopBlocking()
76          {
77              _isBlocking = false;
78          }
79
80          public void IncreaseEnergy(int amount) => Energy += amount;
81
82          public void ResetEnergy() => Energy = 0;
83
84          public virtual void OnDuplicateRecruited()
85          {
86              _transformationState.Handle(this);
87          }
88
89          public void SetTransformationState(ITransformationState
              transformationState)
90          {
91              _transformationState = transformationState;
92          }
93
```

```
 94          public void ResetHealth()
 95          {
 96              _health = _maxHealth;
 97              StopBlocking(); // Ensure blocking is reset when health is
                   reset
 98          }
 99      }
100  }
101
```