```
/* PropertyCell.cs
 * Final Project
 * Revision History
 * Changho Choi, 2015.08.05: Created
*/
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Diagnostics;

namespace Monopoly
{
    class PropertyCell : Cell
    {
        Boolean available;
        int cellPrice;
        int rentPrice;
        Player owner;

        public PropertyCell()
        {
        }

        public PropertyCell(int index, int cellPrice, int rentPrice)
            : base(index, "Property Cell " + index)
        {
            Debug.WriteLine("Property Cell Created with index" + index);

            available = true;
            owner = null;
            this.cellPrice = cellPrice;
            this.rentPrice = rentPrice;
        }

        public override int GetPrice()
        {
            return this.cellPrice;
        }

        public override int GetRentPrice()
        {
            return this.rentPrice;
        }

        public override Boolean IsAvailable()
        {
            return available;
        }

        public override void SetAvailable(Boolean available)
        {
            this.available = available;
        }

        public override Player GetOwner()
        {
            return this.owner;
        }

        public override void SetOwner(Player owner)
        {
            this.owner = owner;
            this.SetAvailable(false);
        }

        public override void LandedOn(Player curPlayer)
        {
            string answer;
            Debug.WriteLine("Player Landed on PropertyCell Cell");
            Console.WriteLine(curPlayer.Name + " arrived at " + this.CellName);
            if (this.available)
            {
                Console.WriteLine("This cell is available");
                Console.Write("Do you want to buy this cell? (y/n): ");
                answer = Console.ReadLine();

                if (answer == "y" || answer == "Y")
                {
                    BuyProperties(curPlayer);  //Refactoring with Extract Method
                }

            }
            else if (this.owner != curPlayer)
            {
                PayRent(curPlayer); //Refactoring with Extract Method
            }

        } //public override void LandedOn(Player curPlayer)

        //Refactoring with Extract Method
        private void PayRent(Player curPlayer)
        {
            Console.WriteLine("Unfortunaetly you are in someone's properties. You have to pay rent for him");
            if (curPlayer.PayRentTo() == true)
            {
                Console.WriteLine("Rent fee $" + this.rentPrice + " have been paid to " + this.owner.Name);
            }
            else
            {
                Console.WriteLine("You dont have suffecient funds!");
                if ((curPlayer.SellProperty() == false) || (curPlayer.PayRentTo() == false))
                {
                    curPlayer.IsKickedOut = true;
                    Console.WriteLine("You dont have any propertie to sell \nYou have been kicked out of the game");
                }
                else
                {
                    Console.WriteLine("Rent fee $" + this.rentPrice + " have been paid to " + this.owner.Name);
                }
            }
        }

        //Refactoring with Extract Method
        private static void BuyProperties(Player curPlayer)
        {
            if (curPlayer.BuyProperty() == true)
            {
                Console.WriteLine("Congratulation , you own this cell now");
            }
            else
            {
                Console.WriteLine("You dont have suffecient money!!. Please sell a property ");
                while (curPlayer.getPropertyNumber() > 0)
                {
                    if ((curPlayer.SellProperty() == true) && (curPlayer.BuyProperty() == true))
                    {
                        Console.WriteLine("Congratulation , you own this cell now");
                        break;
                    }
                    else
                    {
                        Console.WriteLine("Sorry, due to shortage of money, it was not processed!");
                    }
                }
            }
        }
    } // class PropertyCell : Cell
} //namespace Monopoly
```