```
import java.util.HashMap;
import java.util.Map;
import java.util.InputMismatchException;
import java.util.Scanner;
public class Main {
    static class Item {
        private String name;
        private double startingPrice;
        private double highestBid;
        private String highestBidder;
        public Item(String name, double startingPrice) {
            this.name = name;
            this.startingPrice = startingPrice;
            this.highestBid = startingPrice;
            this.highestBidder = "None";
        }
        public String getName() {
            return name;
        }
        public double getStartingPrice() {
            return startingPrice;
        }
        public double getHighestBid() {
            return highestBid;
        }
        public String getHighestBidder() {
            return highestBidder;
        }
        public boolean placeBid(double bidAmount, String bidderName) {
            if (bidAmount > highestBid) {
                highestBid = bidAmount;
                highestBidder = bidderName;
                return true;
            }
            return false;
        }
        @Override
        public String toString() {
            return "Item: " + name + "\nStarting Price: $" + startingPrice +
```

```
            "\nCurrent Highest Bid: $" + highestBid +
            " by " + highestBidder;
    }
}
static class Auction {
    private Map<String, Item> items;
    public Auction() {
        items = new HashMap<>();
    }
    public void addItem(Item item) {
        items.put(item.getName(), item);
    }
    public void placeBid(String itemName, double bidAmount, String bidderName) {
        Item item = items.get(itemName);
        if (item != null) {
            boolean success = item.placeBid(bidAmount, bidderName);
            if (success) {
                System.out.println("Bid placed successfully!");
            } else {
                System.out.println("Bid too low. Try again.");
            }
        } else {
            System.out.println("Item not found.");
        }
    }
    public void showItemDetails(String itemName) {
        Item item = items.get(itemName);
        if (item != null) {
            System.out.println(item);
        } else {
            System.out.println("Item not found.");
        }
    }
    public void showAllItems() {
        if (items.isEmpty()) {
            System.out.println("No items available for bidding.");
        } else {
            items.values().forEach(item -> System.out.println(item));
        }
```

```
        }
        public void endAuction(String itemName) {
            Item item = items.get(itemName);
            if (item != null) {
                System.out.println("Auction for " + item.getName() + " ended!");
                System.out.println("Winning Bid: $" + item.getHighestBid() + " by " +
item.getHighestBidder());
            } else {
                System.out.println("Item not found.");
            }
        }
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Auction auction = new Auction();
        while (true) {
            try {
                System.out.println("\n1. Add Item");
                System.out.println("2. Place Bid");
                System.out.println("3. Show Item Details");
                System.out.println("4. View All Items");
                System.out.println("5. End Auction");
                System.out.println("6. Exit");
                System.out.print("Choose an option: ");
                int choice = getValidIntegerInput(scanner);
                switch (choice) {
                    case 1:
                        System.out.print("Enter item name: ");
                        String name = scanner.nextLine();
                        System.out.print("Enter starting price: ");
                        double startingPrice = getValidDoubleInput(scanner);
                        auction.addItem(new Item(name, startingPrice));
                        System.out.println("Item added successfully.");
                        break;
                    case 2:
                        System.out.print("Enter item name: ");
                        String itemName = scanner.nextLine();
                        System.out.print("Enter bid amount: ");
                        double bidAmount = getValidDoubleInput(scanner);
```

```
                System.out.print("Enter your name: ");
                String bidderName = scanner.nextLine();
                auction.placeBid(itemName, bidAmount, bidderName);
                break;
            case 3:
                System.out.print("Enter item name: ");
                String showItemName = scanner.nextLine();
                auction.showItemDetails(showItemName);
                break;
            case 4:
                auction.showAllItems();
                break;
            case 5:
                System.out.print("Enter item name to end auction: ");
                String endItemName = scanner.nextLine();
                auction.endAuction(endItemName);
                break;
            case 6:
                System.out.println("Exiting...");
                scanner.close();
                return;
            default:
                System.out.println("Invalid option. Try again.");
        }
      } catch (InputMismatchException e) {
        System.out.println("Invalid input. Please enter a valid number.");
        scanner.nextLine();
      } catch (Exception e) {
        System.out.println("An unexpected error occurred: " + e.getMessage());
        e.printStackTrace();
      }
    }
}
private static int getValidIntegerInput(Scanner scanner) {
    int input = -1;
    boolean valid = false;
    while (!valid) {
      try {
        input = scanner.nextInt();
```

```
            scanner.nextLine();
            if (input < 1 || input > 6) {
                throw new IllegalArgumentException("Option must be between 1 and 6.");
            }
            valid = true;
        } catch (InputMismatchException e) {
            System.out.println("Invalid input. Please enter a valid number.");
            scanner.nextLine(); // Consume the invalid input
        } catch (IllegalArgumentException e) {
            System.out.println(e.getMessage());
        }
    }
    return input;
}
private static double getValidDoubleInput(Scanner scanner) {
    double input = -1;
    boolean valid = false;
    while (!valid) {
        try {
            input = scanner.nextDouble();
            scanner.nextLine();
            if (input <= 0) {
                throw new IllegalArgumentException("Amount must be greater than zero.");
            }
            valid = true;
        } catch (InputMismatchException e) {
            System.out.println("Invalid input. Please enter a valid number.");
            scanner.nextLine();
        } catch (IllegalArgumentException e) {
            System.out.println(e.getMessage());
        }
    }
    return input;
}
}
```