```python
# ------------------------TASK 1-------------------------------------

less_than_ten = True
while less_than_ten:
    try:
        numItems = int(input("Enter amount of items to be put on aution " +
                             "(atleast 10): "))
        if numItems >= 10:
            less_than_ten = False
    except ValueError:
        print("\nCan only be numbers.")

# Lists containing different properties of auction items.
# Dictionary would have worked better but O' Level restrictions.
ItmNumList = []
ItmDescList = []
ReservePriceList = []
NumBidsList = []       # List containing number of bids for each item
BidList = []           # List containing highest bid for each item
BuyerNumList = []      # List containing buyer number of highest bidders
SoldList = []          # List containing whether each item is sold or not

# Loop for number of items times.
for i in range(numItems):
    all_input_correct = False
    while not all_input_correct:
        num = input("Enter item number: ")
        try:
            # Check if input can be converted to integer.
            # If it can't be converted then it contains non numbers.
            int(num)

        except ValueError:
            print("Item number may only contain whole numbers.")

        else:
            # This part only executes if num contains only numbers.
            if int(num) < 0:
                # Negative numbers are not allowed.
                print("Item number may only contain whole numbers.")
            elif num in ItmNumList:
                print("Item number needs to be unique.")
            else:
                ItmNumList.append(num)
                all_input_correct = True

    ItmDescList.append(input("Enter item description: "))

    reserve_price_input = input("Enter reserve price: $")
    is_number = False
    # Defaults to false so that condition is checked at least once.
    while not is_number:
        # This part is checked repeatedly until input is valid.
        try:
            int(reserve_price_input)
        except ValueError:
            print("Reserve price may only be a positive whole number." +
                  " Try again.")
```

```
59              reserve_price_input = input("Enter reserve price: $")
60          else:
61              # is_number is set to True only when ValueError is not raised.
62              is_number = True
63
64      # The input does not immediately get added to the reserve prices list.
65      while reserve_price_input < 0:
66          print("Reserve price must be positive. Try again.")
67          reserve_price = int(input("Enter reserve price: $"))
68      ReservePriceList.append(reserve_price_input)  # Add it after the checks.
69
70      NumBidsList.append(0)
71      BidList.append(0)
72      BuyerNumList.append("")
73      SoldList.append(False)
74
75
76
77  # ----------------------------TASK 2---------------------------------------
78
79  # Print all the available items for selection using Item Number.
80  print("Available items:")
81  for i in range(numItems):
82      print(ItmNumList[i], ItmDescList[i], sep=": ")
83
84  WantToBid = True  # When false; break out of loop.
85  while WantToBid:
86      choice = input("Do you want to place a bid? (y/n): ")
87      # If the choice is 'n' then WantToBid is set to False and the elif
88      # segment does not run.
89      # If choice is 'y' then WantToBid is not modified and the elif segment
90      # is run.
91      # If choice is neither 'y' nor 'n' then nothing happens and the user is
92      # prompted again.
93
94      if choice == 'n':
95          WantToBid = False
96
97      elif choice == 'y':
98          SelectedItem = ''  # Stores item number of selected item.
99          BidAmount = 0
100         BuyerNumber = ''
101
102         item_num_correct = False  # True if selected item is available.
103         while not item_num_correct:
104             SelectedItem = input("Enter item number from above: ")
105             if SelectedItem in ItmNumList:
106                 # This segment only executes if the selected item number
107                 # exists in ItmNumList.
108                 item_num_correct = True
109
110                 list_index = ItmNumList.index(SelectedItem)
111                 print()  # Blank line
112                 print(SelectedItem, ItmDescList[list_index])
113                 print("Highest bid: $" + str(BidList[list_index]))
114
115             else:
116                 print("Invalid item number; try again.")
117
```

```
118         bid_correct = False  # True if bid is higher than current highest.
119         while not bid_correct:
120             BidAmount = int(input("Enter your bid: $"))
121             if BidAmount > BidList[ItmNumList.index(SelectedItem)]:
122                 bid_correct = True
123             else:
124                 print("Bid amount must be higher than previous bid.")
125 
126         BuyerNumber = input("Enter buyer number: ")
127         list_index = ItmNumList.index(SelectedItem)  # Index of item.
128 
129         BidList[list_index] = BidAmount
130         BuyerNumList[list_index] = BuyerNumber
131         NumBidsList[list_index] += 1
132 
133 
134 
135 # --------------------------------TASK 3-----------------------------------
136 TotalFee = 0.0  # 0.0 instead of 0 because it needs to be float.
137 LessThanReservePrice = []  # Items with highest bid lower than reserve.
138 NoBids = []  # Items with no bids.
139 
140 for i in range(numItems):
141     if BidList[i] >= ReservePriceList[i]:  # Sold?
142         SoldList[i] = True
143         TotalFee += BidList[i] * 0.1  # Fee is 10% of bid.
144     else:
145         LessThanReservePrice.append(ItmNumList[i])
146     if NumBidsList[i] == 0:  # No bids?
147         NoBids.append(ItmNumList[i])
148 
149 
150 # Printing information.
151 print("\n-----------------------")
152 print("Total fee: " + str(TotalFee))
153 print("Number of items sold: " + str(numItems - len(LessThanReservePrice)))
154 
155 print("\nThe {0} items that have not".format(len(LessThanReservePrice)),
156       "reached their reserved price are:")
157 for x in LessThanReservePrice:
158     print(x, " Highest bid: $", BidList[ItmNumList.index(x)], sep='')
159 
160 print("\nThe {0} items that have recieved no bids are:".format(len(NoBids)))
161 print(', '.join(NoBids))  # Print NoBids delimited with ', '
162 
163 input()  # Wait before exiting.
```