```
# ------------------------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.")
            reserve_price_input = input("Enter reserve price: $")
        else:
            # is_number is set to True only when ValueError is not raised.
            is_number = True

    # The input does not immediately get added to the reserve prices list.
    while reserve_price_input < 0:
```

```
        print("Reserve price must be positive. Try again.")
        reserve_price = int(input("Enter reserve price: $"))
    ReservePriceList.append(reserve_price_input)  # Add it after the checks.

    NumBidsList.append(0)
    BidList.append(0)
    BuyerNumList.append("")
    SoldList.append(False)



# ---------------------------TASK 2----------------------------------------

# Print all the available items for selection using Item Number.
print("Available items:")
for i in range(numItems):
    print(ItmNumList[i], ItmDescList[i], sep=": ")

WantToBid = True  # When false; break out of loop.
while WantToBid:
    choice = input("Do you want to place a bid? (y/n): ")
    # If the choice is 'n' then WantToBid is set to False and the elif
    # segment does not run.
    # If choice is 'y' then WantToBid is not modified and the elif segment
    # is run.
    # If choice is neither 'y' nor 'n' then nothing happens and the user is
    # prompted again.

    if choice == 'n':
        WantToBid = False

    elif choice == 'y':
        SelectedItem = ''  # Stores item number of selected item.
        BidAmount = 0
        BuyerNumber = ''

        item_num_correct = False  # True if selected item is available.
        while not item_num_correct:
            SelectedItem = input("Enter item number from above: ")
            if SelectedItem in ItmNumList:
                # This segment only executes if the selected item number
                # exists in ItmNumList.
                item_num_correct = True

                list_index = ItmNumList.index(SelectedItem)
                print()  # Blank line
                print(SelectedItem, ItmDescList[list_index])
                print("Highest bid: $" + str(BidList[list_index]))

            else:
                print("Invalid item number; try again.")

        bid_correct = False  # True if bid is higher than current highest.
        while not bid_correct:
            BidAmount = int(input("Enter your bid: $"))
            if BidAmount > BidList[ItmNumList.index(SelectedItem)]:
                bid_correct = True
            else:
                print("Bid amount must be higher than previous bid.")

        BuyerNumber = input("Enter buyer number: ")
        list_index = ItmNumList.index(SelectedItem)  # Index of item.

        BidList[list_index] = BidAmount
        BuyerNumList[list_index] = BuyerNumber
```

```
        NumBidsList[list_index] += 1



# ------------------------------TASK 3-----------------------------------
TotalFee = 0.0  # 0.0 instead of 0 because it needs to be float.
LessThanReservePrice = []  # Items with highest bid lower than reserve.
NoBids = []  # Items with no bids.

for i in range(numItems):
    if BidList[i] >= ReservePriceList[i]:  # Sold?
        SoldList[i] = True
        TotalFee += BidList[i] * 0.1  # Fee is 10% of bid.
    else:
        LessThanReservePrice.append(ItmNumList[i])
    if NumBidsList[i] == 0:  # No bids?
        NoBids.append(ItmNumList[i])


# Printing information.
print("\n-----------------------")
print("Total fee: " + str(TotalFee))
print("Number of items sold: " + str(numItems - len(LessThanReservePrice)))

print("\nThe {0} items that have not".format(len(LessThanReservePrice)),
      "reached their reserved price are:")
for x in LessThanReservePrice:
    print(x, " Highest bid: $", BidList[ItmNumList.index(x)], sep='')

print("\nThe {0} items that have recieved no bids are:".format(len(NoBids)))
print(', '.join(NoBids))  # Print NoBids delimited with ', '

input()  # Wait before exiting.
```