Given a class diagram for **`List`** and **`ItemNode`** as in **Figure 1a** and **Figure 1b**. Complete a linked list program given in **Program1**.

Define **`ItemNode`** class and write a constructor for class **`ItemNode`** with 2 default arguments, name equal to null and price equal to 0 – initialize the data members of **`name`**, **`price`**, **`aft`**, and **`bef`** with suitable values. Define **`List`** class and write the implementation for all the function members for **`List`** to accomplish each of the following tasks:

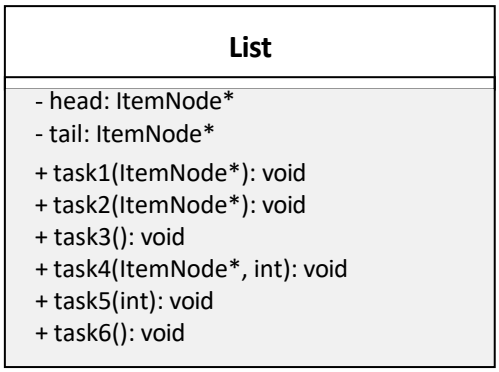

Figure 1a: `List` class

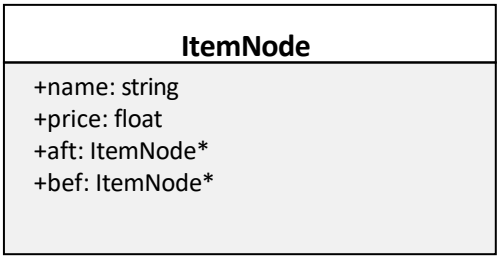

Figure 1b: `ItemNode` class

```
1   //Program 1
2   /*Group Member List
3   Member Name 1:
4   Matric Number:
5   Section:
6
7   Member Name 2:
8   Matric Number:
9   Section:
10
11  ...
12  ...
13  */
14
15  int main()
16  {
17     List ItemList;
18     ItemNode *n1 = new ItemNode("Book", 9.59);
19     ItemNode *n2 = new ItemNode("Ruler", 3.45);
20     ItemNode *n3 = new ItemNode("Pen", 5.69);
21     ItemNode *n4 = new ItemNode("Eraser", 2.25);
22
23     cout << fixed << setprecision(2) << left;
24     ItemList.task1(n1);
25     ItemList.task2(n2);
26     ItemList.task3();
27     ItemList.task4(n3, 2);
28     ItemList.task4(n4, 2);
29     ItemList.task3();
30     ItemList.task5(3);
31     ItemList.task3();
32     ItemList.task6();
33     ItemList.task3();
34
35     return 0;
36  }
```

Task 1: Define a function named **`task1`**. The function should able to insert a new item node into an empty list. **Figure 2** shows the result of this task after the execution of the statement in **Line 24**. Illustrate the step-by-step

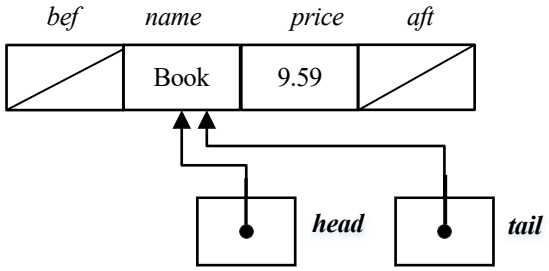

Figure 2: Current node in the list after insertion node in the empty list

Task 2: Define a function named **`task2`**. The function should able to insert a new item node in front of the list. **Figure 3** shows the result of this task after the execution of the statement in **Line 25**.

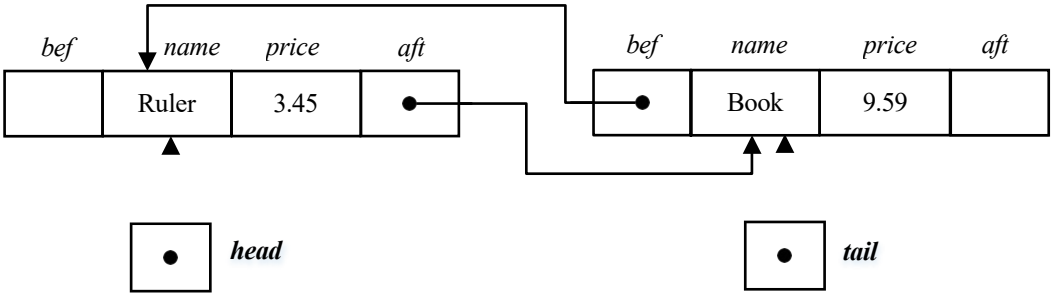

Figure 3: Current nodes in the list after the first insertion node at the end of the list

Task 3: Define a function named **`task3`**. The function should able to print all data in the list in backward order. **Figure 4** shows the result of this task after the execution of the statement in **Line 26**. ***Note:*** In this task, you should use a **loop**.

```
Print backward:
[Book    9.59]  [Ruler   3.45]
```

Figure 4: Output after the execution of the statement in Line 26

Task 4: Define a function named **`task4`**. The function should able to insert a new item node at the middle of the list. The position of the new node in the list depends on the **value of the second argument** of this function. Assume the value of the second argument of this function is 2. This means that the new node will be the second node in the list (being in the second position in the list). **Figure 5** shows the result of this task after the execution of the statement in **Line 27** and **Figure 6** shows the result of this task after the execution of the statement in **Line 28.** ***Note:*** In this task, you should use a **loop**.

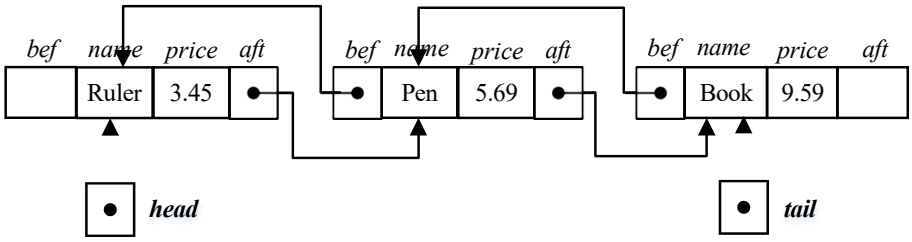

Figure 5: Current nodes in the list after the first insertion node at the second position in the list

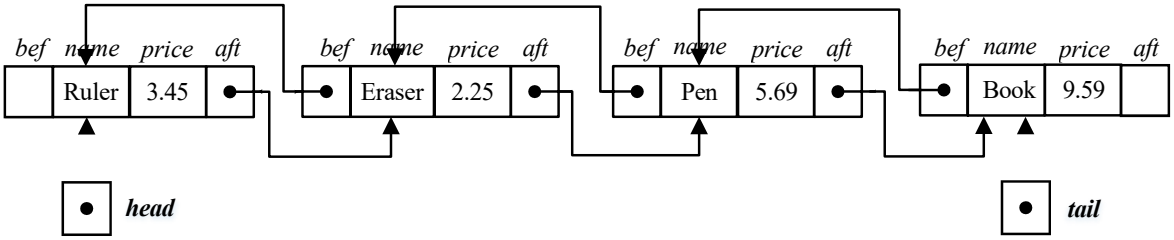

Figure 6: Current nodes in the list after the second insertion node at the second position in the list

Task 5: Define a function named **`task5`**. The function should able to delete the item node at the middle of the list. The position of the node to delete in the list depends on the **value of the second argument** of this function. Assume the value of the second argument of this function is 2. This means that the node that will be deleted is the second node in the list. **Figure 7** shows the result of this task after the execution of the statement in **Line 30**. ***Note:*** In this task, you should use a **loop**.

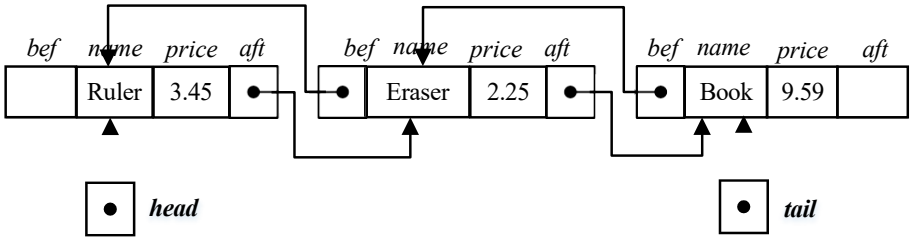

Figure 7: Current nodes in the list after the deletion of the second node in the list

Task 6: Define a function named **`task6`**. The function should able to delete the last node in the list. **Figure 8** shows the result of this task after the execution of the statement in **Line 32**.

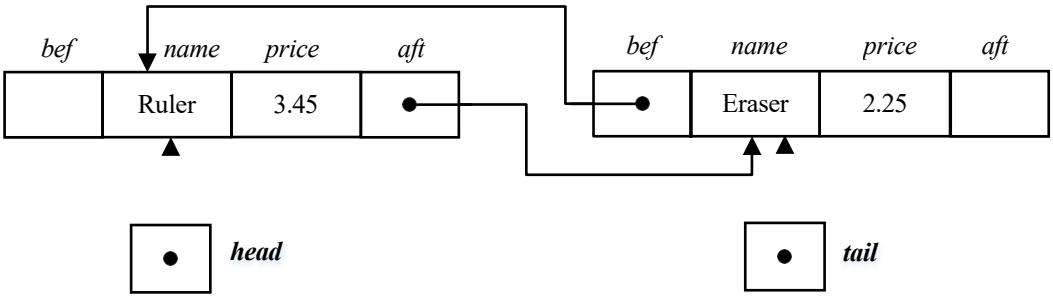

Figure 8: Current nodes in the list after the deletion of the last node in the list

After completing all tasks, please make sure your program should be able to produce the output as shown in **Figure 9.**

```
Print backward:
[Book    9.59]  [Ruler   3.45]

Print backward:
[Book    9.59]  [Pen     5.69]  [Eraser  2.25]  [Ruler   3.45]

Print backward:
[Book    9.59]  [Eraser  2.25]  [Ruler   3.45]

Print backward:
[Eraser  2.25]  [Ruler   3.45]
```

Figure 9: The output of completed Program 1