

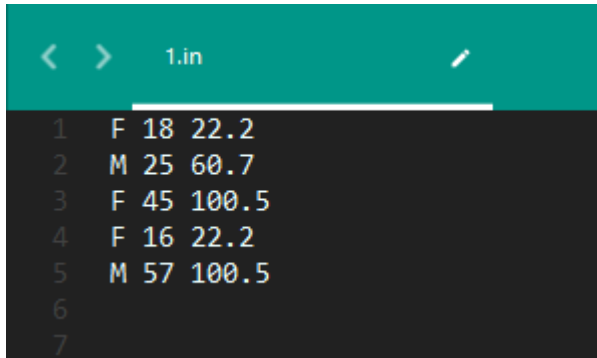
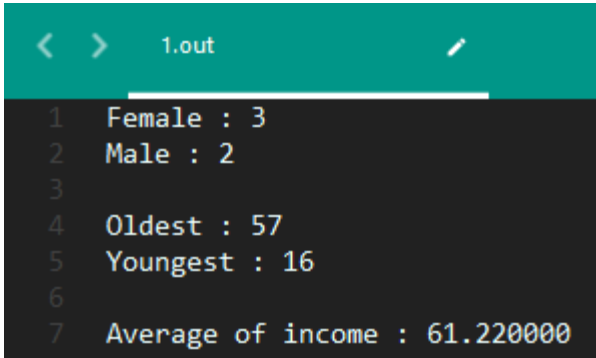
Introduction to Computers and Programming LAB-11_{2016/12/21}

- ✧ The output must be in our sample output format.
- ✧ You can raise your hand to demo once you finish a program.
- ✧ TAs will update lab records every Monday after the lab hours in the link: <http://goo.gl/ZVJu2Y>

1. Dating Agency

A dating agency is going to analyze the registration of a group dating. The registration forms have already been aggregated into a file, which contains the list of applicants. Each line represents an applicant in a format: Gender (*M/F, char*), Age (*0~100, integer*), Income (*0~22000, double*).

Please write a program to analyze the file of the dating list. **Inputs are input/output filenames, the program reads input file, and writes the following information to output file: (1) the number of F/M applicant ;(2) the oldest/youngest applicant (3) the average of income.**

Program Window																																																	
<pre>Input filename: 1.in Output filename: 1.out Analyze successfully! Input filename:</pre>																																																	
Input/Output file example																																																	
 <p>The screenshot shows a text editor window titled '1.in'. It contains five lines of data, each with a line number (1-5) followed by a space, a gender character (F or M), a space, an age integer, a space, and an income double value. Line 6 and 7 are empty.</p> <table><tr><th>Line</th><th>Gender</th><th>Age</th><th>Income</th></tr><tr><td>1</td><td>F</td><td>18</td><td>22.2</td></tr><tr><td>2</td><td>M</td><td>25</td><td>60.7</td></tr><tr><td>3</td><td>F</td><td>45</td><td>100.5</td></tr><tr><td>4</td><td>F</td><td>16</td><td>22.2</td></tr><tr><td>5</td><td>M</td><td>57</td><td>100.5</td></tr><tr><td>6</td><td></td><td></td><td></td></tr><tr><td>7</td><td></td><td></td><td></td></tr></table>	Line	Gender	Age	Income	1	F	18	22.2	2	M	25	60.7	3	F	45	100.5	4	F	16	22.2	5	M	57	100.5	6				7				 <p>The screenshot shows a text editor window titled '1.out'. It contains seven lines of summary statistics, each with a line number (1-7) followed by a space and the text. Line 3 is empty.</p> <table><tr><th>Line</th><th>Text</th></tr><tr><td>1</td><td>Female : 3</td></tr><tr><td>2</td><td>Male : 2</td></tr><tr><td>3</td><td></td></tr><tr><td>4</td><td>Oldest : 57</td></tr><tr><td>5</td><td>Youngest : 16</td></tr><tr><td>6</td><td></td></tr><tr><td>7</td><td>Average of income : 61.220000</td></tr></table>	Line	Text	1	Female : 3	2	Male : 2	3		4	Oldest : 57	5	Youngest : 16	6		7	Average of income : 61.220000
Line	Gender	Age	Income																																														
1	F	18	22.2																																														
2	M	25	60.7																																														
3	F	45	100.5																																														
4	F	16	22.2																																														
5	M	57	100.5																																														
6																																																	
7																																																	
Line	Text																																																
1	Female : 3																																																
2	Male : 2																																																
3																																																	
4	Oldest : 57																																																
5	Youngest : 16																																																
6																																																	
7	Average of income : 61.220000																																																

2. Find different words

Please write to calculate how many different words in given files. (Please Ignore case)

(There will no more than 1000 different words, and max length of word is 100)

Input:

1. How many files
2. Path of each file

Output:

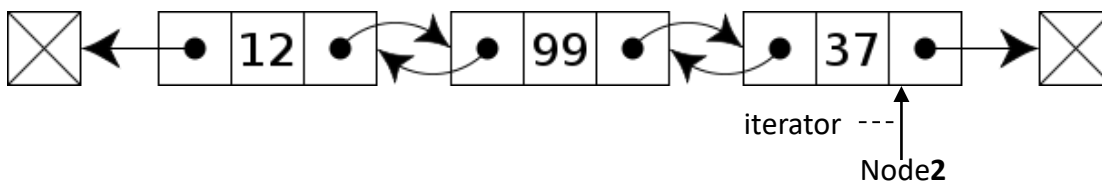
How many different words in those files.

Example:

```
How many files would you input: 2
Please input file path: text1.txt
Please input file path: text2.txt
There are 24 different string in those files.
```

3. Doubly linked list

A doubly linked list is a linked data structure that consists of a set of sequentially linked records called nodes. Each node contains two fields, called links, that are references to the previous and to the next node in the sequence of nodes.



Design a doubly linked list that contains a special pointer called an iterator, that can tell which of the node is the iterator points to. Functions of it are described as below:

- (1) ***insert_left***: insert a node with the input value at the left of the node which the iterator points to.
- (2) ***insert_right***: insert a node with the input value at the right of the node which the iterator points to.
- (3) ***move_left***: move the iterator to the left of the node which the iterator pointed to.
- (4) ***move_right***: move the iterator to the right of the node which the iterator pointed to.
 - If the left node (or right node), or even the node is null, just print “Cannot move to the left node (or right node)”
- (5) ***remove_node***: remove the node which the iterator points to, and then move the iterator to the left node. If the node didn't have left node, then move the iterator to the right node. If the node didn't have both left and right node, then the iterator will not point to any node after the removal, because the list will be empty after the removal.
- (6) ***print_list***: print the whole linked list.

Note:

- (1) You need to print the iterator (Node*i* which it points to and the value of the Node*i*) after each function finished its work (even if the function didn't work), expect for the situation that the list is empty.
- (2) The Node is start form Node0, which is the leftmost Node.
- (3) After insert_left/insert_right, the iterator should be pointed to the same node before the insertion.
- (4) If the list is empty, then after insertion, the iterator must be pointed to the first node which is Node0.

Examples:

```
Functions:(1)insert_left (2)insert_right (3)move_left  
(4)move_right (5)remove_node (6)print_list
```

1.

```
Select functions:1  
Input the value:50  
Iterator:Node0 Value:50  
  
Select functions:1  
Input the value:49  
Iterator:Node1 Value:50  
  
Select functions:1  
Input the value:48  
Iterator:Node2 Value:50  
  
Select functions:2  
Input the value:51  
Iterator:Node2 Value:50  
  
Select functions:2  
Input the value:52  
Iterator:Node2 Value:50  
  
Select functions:6  
List:49->48->50->52->51  
  
Iterator:Node2 Value:50
```

2.

```
Select functions:3  
Iterator:Node1 Value:48  
  
Select functions:3  
Iterator:Node0 Value:49  
  
Select functions:3  
Cannot move to the left node  
Iterator:Node0 Value:49  
  
Select functions:5  
Iterator:Node0 Value:48  
  
Select functions:5  
Iterator:Node0 Value:50  
  
Select functions:6  
List:50->52->51
```

3.

```

Iterator:Node0 Value:50
Select functions:4
Iterator:Node1 Value:52
Select functions:4
Iterator:Node2 Value:51
Select functions:4
Cannot move to the right node
Iterator:Node2 Value:51
Select functions:5
Iterator:Node1 Value:52
Select functions:6
List:50->52
Iterator:Node1 Value:52

```

4.

```

Select functions:5
Iterator:Node0 Value:50
Select functions:5
Select functions:5
Cannot remove node from empty list
Select functions:6
Empty list
Select functions:4
Cannot move to the right node
Select functions:3
Cannot move to the left node

```

4. Sum of sequence

Given an integer $3 \leq n \leq 15$, represents the sequence $1, 2, \dots, n-1, n$ will be generated. Use these 3 operators “ ” (concatenate two numbers), “+” (sum up two numbers) and “-” (subtract two numbers) on each two numbers in the sequence, then **find out and write the equations that generate the summation 0, and the number of equations that equal to 0 to the screen and the file “n.txt”**.

Here are some examples to make you clearer. For input $n = 7$, the equations below demonstrate how to use these 3 operators:

(1) $1 + 2 + 3 - 4 - 5 - 6 + 7 \Rightarrow 1+2+3-4-5-6+7 = -2$

(2) $1 \quad 2 + 3 - 4 + 5 \quad 6 \quad 7 \Rightarrow 12+3-4+567 = 578$

(3) $1 - 2 \quad 3 - 4 \quad 5 + 6 \quad 7 \Rightarrow 1-23-45+67 = 0 \Rightarrow \text{print/write } 1 - 2 \quad 3 - 4 \quad 5 + 6 \quad 7$

You can check your result using the following form.

Input	3	4	5	7	8	10
# of equation = 0	1	1	1	6	10	17

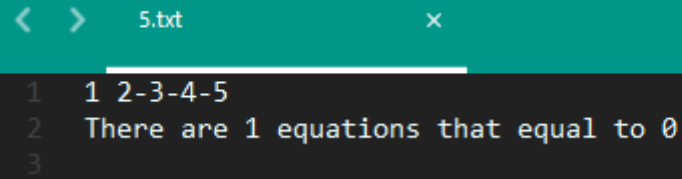
Program Window

```
Input a number: 5
There are 1 equations that equal to 0

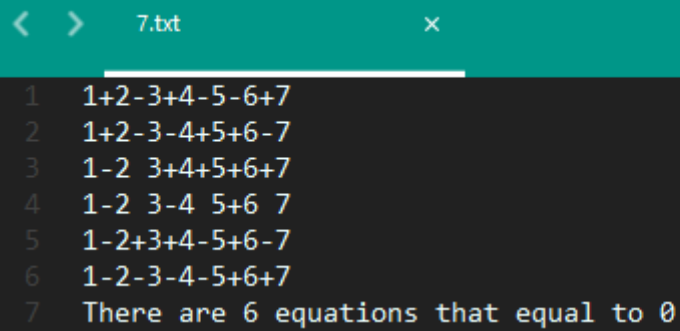
Input a number: 7
There are 6 equations that equal to 0

Input a number:
```

Output file



```
< > 5.txt x
1 1 2-3-4-5
2 There are 1 equations that equal to 0
3
```



```
< > 7.txt x
1 1+2-3+4-5-6+7
2 1+2-3-4+5+6-7
3 1-2 3+4+5+6+7
4 1-2 3-4 5+6 7
5 1-2+3+4-5+6-7
6 1-2-3-4-5+6+7
7 There are 6 equations that equal to 0
```