You do not need to consider exceptions that are not mentioned in this document.
You have to print exactly same as sample outputs.

1. Phone Book

1-1. constraints

- Always show prompt CP-2019-12345> (your student ID) before each task.
- Initial state: A state that can enter commands (1. Add person/2. Remove person/3. Print phone book)
- In the initial state, when the user inputs empty line (just \n), it shows information about choices.
- In the initial state, when the user inputs exit, end the program.
- When each menu is finished, it returns to the initial state to wait another input of the user.

1-2. sample input & output

<table>
<thead>
<tr>
<th>CP-2019-12345&gt; (input enter)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phone Book</td>
</tr>
<tr>
<td>1. Add person</td>
</tr>
<tr>
<td>2. Remove person</td>
</tr>
<tr>
<td>3. Print phone book</td>
</tr>
<tr>
<td>CP-2019-12345&gt;</td>
</tr>
</tbody>
</table>

2. Add person

2-1. constraints

- User can add person from the Add person menu.
- In the initial state, when the user inputs 1, it enters the Add person menu and shows information about choices.
- Each person stores his/her first, last name and phone number.
- There must be a space between the first and last names.
- User inputs only 02-xxxx-xxxx or 010-xxxx-xxxx format as phone number.
- Person who is categorized in Work stores his/her team.
- Person who is categorized in Family stores his/her birthday.
- User inputs only YYMMDD format as birthday.
- Person who is categorized in Friend stores his/her age.
- After the task is done, print Successfully added new person.

2-2. sample input & output
3. Remove person

3-1. constraints

- User removes information of person from the Remove person menu.
- In the initial state, when the user inputs 2, it enters the Remove person menu and asks index of person to remove.
- example of index policy
  - In the phone book with only one person, if user remove a person whose index is 1 and adds another person, the index of new person becomes 1.

3-2. sample input & output

//If the index is available
CP-2019-12345>2
Enter index of person: 10
A person is successfully deleted from the Phone Book!
CP-2019-12345>

//If not
CP-2019-12345>2
Enter Index of person: 10
Person does not exist!
CP-2019-12345>

4. Print person

4-1. constraints
User can print all the stored people and their information.

In the initial state, when the user inputs 3, it prints information of all persons.

People who have been removed should not print.

Output format is

- Person class
  
  {first name} {last name}_{phone number}

- Work, Friend
  
  {first name} {last name}_{phone number}_{an additional attribute}

- Family
  
  {first name} {last name}_{phone number}_{birthday}_{D-day}

4-2. Sample input & output

```plaintext
CP-2019-12345>3
Phone Book Print
1. John doe_010-1234-5678_Warriors
2. Stephen Curry_02-1234-5678_940101_261
...
...
CP-2019-12345>
```

5. Exit

5-1. Example

```plaintext
CP-2019-12345>exit
```

6. Report

6-1. What your report should contain

- Implementation Environment
- Specific explanation about the code
- Troubleshooting points while implementing your code
- Screenshot of the program working

7. FAQ

- `exit` must work on 'initial state' only. Otherwise, it is treated as an string 'exit'.
  - For example, when you enter `exit` while entering the first & last name of person(ex: exit exit), that person's first name will be `exit exit`.
  - Also, you can expect no `exit` will be entered while entering phone number, as it is an int type.

- At Initial state, any commands(add person, remove person, print phonebook) should work, regardless of printing command list.

- For Birthday, we do not consider a leap year. A year is same as 365 days.

- For Dday, you must calculate the number of days remaining from the present to the nearest birthday.
  - example) birthday: 940322, today: 3/23 -> D-day: 364 (regardless of leap year)
  - example2) birthday: 940322, today: 3/21 -> D-day: 1 (regardless of leap year)