SPECIFICATIONS

DASTRUC 3rd MACHINE PROBLEM (MP3)

33% of the Final Project Grade

I. INTRODUCTION

This document contains the specifications of the 3rd Machine Problem. **MAKE SURE THAT YOU UNDERSTAND AND FOLLOW THE SPECIFICATIONS PROPERLY. FAILURE TO DO SO WILL AUTOMATICALLY FAIL THIS MP.**

II. OBJECTIVES OF THE MP

a. To apply what you have learned about linked lists, from initialization to operations
b. To apply some linked list operations

III. PROBLEM DESCRIPTION

A local orphanage commissioned you to write a PyCharm project to help determine the gifts to be given to the children this year during the holiday season. You are given the following attributes for each child:

a. First Name: At most 15 characters (no spaces in between)
b. Last Name: At most 15 characters (no spaces in between)
c. Gender: Either 'M' for Male, 'F' for Female
d. Age: Integer
e. Good Deeds: Integer greater than or equal to 0
f. Bad Deeds: Integer greater than or equal to 0

Collectively, we refer to these information as a ***child record***.

Child records are stored in a text file with the name "**INPUT.TXT**". In this machine problem, we will assume that **INPUT.TXT** already exists, and it contains valid values.

A representative example of INPUT.TXT contents for six children are shown below:

```
MARIA SANTOS      F    5    6    1
JOSEFINA CRUZ     F    6    12   5
PEDRO PENDUKO     M    12   16   10
MEDUSA GORGON     F    10   8    2
IOSEF CARL        M    11   0    200
EMIKO SANTOS      F    2    5    0
```

There are six (6) child records in the example. The first line indicates the child's first name (MARIA), last name (SANTOS), gender (F), age (5), good deeds (6), and bad deeds (1) respectively. The remaining lines indicate the same sequence of attribute values for the remaining 5 child records.

The gift to be given to a child is determined based on the values of the last four (4) attributes. The following rules apply:

1. Children up to 5 years of age receive gifts based on the following rules:

Age	Gift for Male	Gift for Female
1	Blue_Pacifier	Pink_Pacifier
2	Colored_Shapes	Colored_Shapes
3	Choo_Choo_Train	Teddy_Bear
4	Wooden_Horse	Doll
5	Remote_Controlled_Car	Pair_of_Shoes

2. For children between ages of 6 and 12 years
   2.1. Compute $Deeds = Good\ Deeds - Bad\ Deeds$
   2.2. If $Deeds \leq 0$, then the gift for the child is "Good_Manners_and_Right_Conduct_Book". Otherwise, the gift is determined based on the following table:

Age	Gift for Male	Gift for Female
6 to 8	Chess_Set	Disney_Puzzle
9 to 12	Soccer_Ball	Blouse_Pants

3. For children more than 10 years old, the gift is an "E_Christmas_Card"

The output of the program is a text file named "**OUTPUT.TXT**", which stores a list of the following:

a. Last Name
b. First Name
c. Gender
d. Gift

The list should be stored alphabetically (ascending, from A to Z) by last name. Using the example input file and the rules for determining the gifts, the output file OUTPUT.TXT will contain:

```
CARL IOSEF          M     Good_Manners_and_Right_Conduct_Book
CRUZ JOSEFINA       F     Disney_Puzzle
GORGON MEDUSA       F     Blouse_Pants
PENDUKO PEDRO       M     Soccer_Ball
SANTOS EMIKO        F     Colored_Shapes
SANTOS MARIA        F     Pair_of_Shoes
```

For this machine problem, we define `ChildProfile` below. The class members correspond to the attributes mentioned in this section (Section III), plus a member corresponding to the gift to be given to a child. No edits should be made in this class. **Edits in this class will incur a –5 point reduction (for each edit) in your total score**.

```
# DO NOT EDIT
class ChildProfile:
    first_name: str
    last_name: str
    gender: str
    age: int
    good_deeds: int
    bad_deeds: int
    gift: str
# END DO NOT EDIT
```

The `ChildProfileList` is the **linked list** implementation of the child record list. All function requirements should be implemented here. You can add variables inside the class, but you cannot add or delete methods (functions) inside the class. **Failure to do so will incur a –5 point reduction (for every added or deleted method) in your total score. Using a Python list implementation will automatically fail this MP.**

```
class ChildProfileList:
    def build_list_from_file(self):
        # <INSERT CODE HERE>

    def swap(self, node_1, node_2):
        # <INSERT CODE HERE>

    def sort_list(self):
        # <INSERT CODE HERE>

    def determine_gift(self):
        # <INSERT CODE HERE>

    def write_gift_file(self):
        # <INSERT CODE HERE>

    # <INSERT NECESSARY CLASS VARIABLES HERE>
```

You will be provided a Python script named "**mp3_classes.py**" containing the `ChildProfile` and `ChildProfileList` class templates. The script also contains the function templates that need to be implemented.

```python
# DO NOT EDIT
class ChildProfile:
    first_name: str
    last_name: str
    gender: str
    age: int
    good_deeds: int
    bad_deeds: int
    gift: str
# END DO NOT EDIT

class ChildProfileList:
    def build_list_from_file(self):
        # <INSERT CODE HERE>

    def swap(self, node_1, node_2):
        # <INSERT CODE HERE>

    def sort_list(self):
        # <INSERT CODE HERE>

    def determine_gift(self):
        # <INSERT CODE HERE>

    def write_gift_file(self):
        # <INSERT CODE HERE>

    # <INSERT NECESSARY CLASS VARIABLES HERE>
```

For the main entry point of the program (i.e., what to call when running the project in PyCharm), the Python script should be called "**main.py**". What is required for this script will be discussed in Section IV.

```python
if __name__ == '__main__':
    # <INSERT CODE HERE>
```

IV. REQUIREMENTS

You are required to implement several functions described below. **NOTE THAT CHANGING THE FUNCTION NAME AND PARAMETERS (ADDING, SUBTRACTING, AND/OR CHANGING THE NAMES OF THE PARAMETERS) WILL AUTOMATICALLY LOSE THE POINTS FOR THAT FUNCTION.**

Function #1:

`def build_list_from_file(self):`

This function is in charge of building the list of child records from **INPUT.TXT**. The specific tasks that this function will do are as follows:

a. Open and read the input file "INPUT.TXT";

b. Store the child record as an element in the list;

c. Repeat step (b) while there are more records from the input file;

d. Close the input file

Function #2:

`def swap(self, node_1, node_2):`

This function is in charge of swapping two child records. The parameters `node_1` and `node_2` represents the respective nodes of the child records to be swapped.

Function #3:

`def sort_list(self):`

This function is in charge of sorting the list in alphabetical order by the child's last name. For children with the same last name, sorting will also be dependent on the child's first name. *In the example above, two children have "SANTOS" as their last name*.

Function #4:

`def determine_gift(self):`

This function is in charge of updating the list of child records by determining and assigning the value of the `gift` member. Refer back to Section III for the rules for determining gifts.

Function #5:

`def write_gift_file(self):`

This function is in charge of writing data into the output file "OUTPUT.TXT". The specific tasks of this function will do are as follows:

a. Create (or open if it exist) the output file "OUTPUT.TXT";

b. For each child record, write a gift record (with the format shown in Section III) on the output file;

c. Repeat step (b) while there are more records in the list;

d. Close the output file

For the Python script **main.py**,

The following tasks will be performed inside main.py:

a. Call the function for building the list of child records;

b. Call the function for sorting the list;

c. Call the function for determining the gift for each child;

d. Call the function for writing data onto the output file;

V. DELIVERABLES AND SUBMISSION DEADLINE

You need to submit **TWO (2)** items, namely:

a. Your Python script for the class files (filename should be <*lastname*>_mp3_classes.py)
b. Your Python script for the main script (filename should be <*lastname*>_main.py)

It is the student's responsibility to check the solution **(several times!)** for syntax and semantic errors before submission. Once submitted, no replacement will be accepted.

The source code should be **RECEIVED** via e-mail as a .zip file (along with the other MP files) before **June 19, 2023, 12:00 noon**. The recipient of your e-mail should be one of the following, depending on your DASTRUC professor. Make sure to CC your and your groupmate's e-mail addresses before sending the e-mail.

John Raymound Javier: jljavier@nu-fairview.edu.ph

Jian Michael Wu: jowu@nu-fairview.edu.ph

The e-mail subject should be:

DASTRUC_<section>:[space]<LASTNAME_MP1>[space]<FIRSTNAME_MP1>[space]MP1_<LASTNAME_MP2>[space]<FIRSTNAME_MP2>[space]MP2_<LASTNAME_MP3>[space]<FIRSTNAME_MP3>[space]MP3

where:

- LASTNAME_MP1 and FIRSTNAME_MP1 = the one who was assigned MP1
- LASTNAME_MP2 and FIRSTNAME_MP2 = the one who was assigned MP2
- LASTNAME_MP2 and FIRSTNAME_MP3 = the one who was assigned MP3
- You should replace [space] with " " (one space bar press)

Ex. DASTRUC_IT123: PEÑA_RYAN_MP1_CANLAS_JOMARY_MP2_LO_HANS_MP3

INCOMPLETE SUBMISSION and **NON-COMPLIANCE** with the instructions and specifications will automatically result in point deductions.

LATE SUBMISSIONS WILL BE ACCEPTED but with a deduction of ONE point for every minute or a fraction thereof in fairness to all students who submitted on time.

VI. CHECKING, TESTING, AND GRADING SCHEME

Your Python Scripts will be compiled and tested. Note that by default you will receive a perfect grade of 100% (for this MP only). It will remain as 100% if there are no warnings, syntax errors, and semantic errors; and that you complied properly with all the MP specifications. **A program with a syntax error will automatically be given a grade of ZERO (0).**

The contribution, in percentage, for each of the required functions are indicated in the table below:

Requirements	Percentage
`def build_list_from_file(self)`	25
`def swap(self, node_1, node_2)`	5
`def sort_list(self)`	20
`def determine_gift(self)`	25
`def write_gift_file(self)`	20
`main.py`	5

** Please see your professor if you have any questions regarding this MP **