Task 1

A student is collating orders for her class for lunch during Teacher's Day celebration:

- Each student has ordered at least 2 items and up to 3 items from the menu.
- Each menu item has a unique ID code. For mains, the ID begins with an 'M'. For sides, the ID begins with an 'S". For drinks and dessert, the ID begins with a 'D'.

You are required to finish setting up the spreadsheet to tabulate the orders and total cost.

Open the file **TASK1.xlsx**. You will see the following data.

Save the file as **TASK1**_<your name>_<class>_<index number>**.xlsx**

Lunch Orders for Teacher's Day Celebration

Orders

No.	Student	Order 1	Cost	Order 2	Cost	Order 3	Cost	Discount	Total Cost
1	Prakash Gibson	D07		D03		S07			
2	Ophelia Eliot	D06		S07		S06			
3	Matthias Aldenberg	M07		D03		D08			
4	Akemi Stanford	M08		M04		D07			
5	Winnie Holt	M09		S04		S07			
6	Justina Kanzaki	M09		S06		D02			
7	Meagan Pender	D01		S05					
8	Sree Kendall	D02		S02		D08			
9	Jody Miyazaki	M08		D07		S06			
10	Dinah Hagen	D07		M04		D02			
11	Naruhito Lewis	D01		M05					
12	Dawn Janssen	M06		S03		D01			
13	Phemie Dries	D06		D03		M07			
14	Javed Leighton	D08		D06					
15	Liddy Pryor	M09		D08		D05			
16	Yūdai Aaltink	D09		M06					
17	Cortney Hermanson	S03		S08		D06			
18	Usman Dustin	S01		D08		D04			
19	Hartley Lam	D07		S06		S09			
20	Lilibeth Nelissen	M09		S02					
21	Kaylynn Cobb	S06		M04		D02			
22	Wynne Field	M05		D09		S08			
23	Veva Constable	S08		S05		M01			
24	Winnifred Richard	S03		M06		D06			
25	Akira Hubert	S09		S08		D09			
							Average Payment:		

Summary

ID	Item Name	Price	Quantity
M01	Tenderloin Steak	$11.30	
M02	Burger Steak	$10.40	
M03	Spring Chicken	$8.30	
M04	Bratwurst	$11.50	
M05	Grilled Salmon Fillet	$12.80	
M06	Fish and Chips	$14.00	
M07	Crayfish Platter	$13.90	
M08	Margherita Pizza	$12.70	
M09	Carbonara	$14.90	
S01	Baked Potatoes	$6.10	
S02	Chili Cheese Fries	$6.70	
S03	Caesar Salad	$6.20	
S04	Onion Rings	$6.70	
S05	Buffalo Wings	$7.10	
S06	Calamari	$6.80	
S07	Lobster Bisque	$7.80	
S08	Cream of Mushroom	$6.00	
S09	Garlic Bread	$6.30	
D01	Hot Cocoa	$6.10	
D02	Black Tea	$5.60	
D03	Cappuccino	$7.00	
D04	Lemonade	$6.60	
D05	Root Beer	$6.70	
D06	Soda Water	$5.40	
D07	New York Cheesecake	$5.00	
D08	Brownie	$5.00	
D09	Apple Strudel	$7.90	

For the following questions, where appropriate, all currency values should be shown in 2 decimal places, e.g. $1.00.

1 In cells **D4 to D28**, use an appropriate function to determine the corresponding prices of each order item in cells **C4 to C28** from the '**Summary**' table on the right. Likewise, complete cells **F4 to F28** with prices of order items in cells **E4 to E28**. [2]

2 In cells **H4 to H28**, use an appropriate formula to determine the corresponding prices of each order item in cells **G4 to G28** from the '**Summary**' table on the right. If the student has not ordered a third item, then the price should be shown as $0.00. [1]

3 If a student has ordered a main, a side and a drink or dessert, then it is considered a **set meal** and the student will be entitled to a **10% discount**.

In cells **I4 to I28**, use an appropriate function to determine the discount amount each student is entitled to, in dollars.

[2]

4 In cells **J4 to J28**, use an appropriate formula to determine the total cost that each student needs to pay for their lunch, taking into consideration the discount that they are each entitled to.

[1]

5 In cells **O4 to O30**, use an appropriate function to determine the total quantity ordered by the class for each menu item.

[2]

6 In cell **J30**, use an appropriate function to determine the average payment to be made by each student.

[1]

7 In cells **J4 to J28**, apply conditional formatting to underline the values that are equal to or greater than 120% of the average value found in **Question 6**.

[1]

Task 2

The following program helps the user to solve a quadratic equation, $ax^2 + bx + c = 0$, where the values for a, b and c are integers input by the user.

```
a = int(input("Enter value of 'a': "))

b = int(input("Enter value of 'b': "))

c = int(input("Enter value of 'c': "))

sol1 = (-b+(b**2-4*a*c)**0.5)/(2*a)

sol2 = (-b-(b**2-4*a*c)**0.5)/(2*a)

print(sol1, sol2)
```

Open the file **TASK2.py**

Save the file as **TASK2_Q8_**<your name>_<class>_<index number>**.py**

8 Edit the program so that it:

(a) Outputs the statement `"The solutions to this quadratic equation are X and Y."` at the end of the program, where `X` and `Y` represents the value of the two solutions. [1]

(b) Checks whether the two solutions are the same. If they are, the program should output the statement `"The only solution to this quadratic equation is X."` at the end instead. [2]

(c) It checks whether the determinant, $b^2 - 4ac$, is less than zero. If it is less than zero, the program should output the statement `"There are no real solutions to this quadratic equation."` without performing any further calculations. [2]

Save your program.

9 Save your program as **TASK2_Q9_**<your name>_<class>_<index number>**.py**

Edit the program so that it:

(a) Asks the user for confirmation after ***a***, ***b*** and ***c*** values are input. For example, `"The quadratic equation is Ax^2 + Bx + C = 0, correct? (Y/N):"` where `A`, `B` and `C` represents the ***a***, ***b*** and ***c*** values keyed in by the user.

If the user inputs `"N"`, then the program should ask to re-enter the values again until the user confirms with a `"Y"`. [2]

(b) When asking the user for confirmation in part **(a)**, the quadratic equation displayed is simplified. The two examples below describe how it should be simplified.

When values of ***a*** and ***b*** are `1` or `-1`, the program should omit the "`1`" character in the equation. For example:

`"-1x^2 + 1x + 2 = 0"` is simplified as `"-x^2 + x + 2 = 0"`.

When values of ***b*** and ***c*** are negative, the program should omit the "`+`" character in the equation. For example:

`"2x^2 + -3x + -4 = 0"` is simplified as `"2x^2 - 3x - 4 = 0"`.

[3]

Save your program.

Task 3

The following program provides information about the COVID-19 vaccination programme in Singapore. The program does the following:

- Allows the user to input their year of birth and whether they have drug allergy.
- Informs the user the type of vaccine they are eligible for and whether an appointment is required, based on the table below:

Age (Years)	No Drug Allergies		With Drug Allergies
<12	Not Eligible		Not Eligible
12 - 17	Pfizer-BioNTech (With appointment)		Not Eligible
18 - 59	Pfizer-BioNTech (With appointment)	Moderna (Without appointment)	Novavax (With appointment)
> 59	Pfizer-BioNTech (Without appointment)	Moderna (Without appointment)	Novavax (Without appointment)

There are several syntax errors and logical errors in this program:

```
age = int(input("Enter your year of birth: ")) - 2021
allergy = int(input("Do you have any drug allergies? (Y/N): "))
eligible_with_appt = ""
eligible_without_appt = ""
if age >= 59:
    if allergy == "N":
        eligible_without_appt += ["Pfizer-BioNTech"]
        eligible_without_appt += ["Moderna"]
    else:
        eligible_without_appt += ["Novavax"]
if age >= 18:
    if allergy == "N":
        eligible_with_appt += ["Moderna"]
        eligible_without_appt += ["Pfizer-BioNTech"]
    else:
        eligible_with_appt += ["Novavax"]
if age > 12:
    if allergy == "Y":
        eligible_with_appt += ["Pfizer-BioNTech"]
if len(eligible_without_appt) ==0 or len(eligible_with_appt) ==0:
    print("You are not eligible for any vaccination.")
else:
print("You are eligible for:")
for vaccine in eligible_without_appt:
    print("- {} vaccine without appointment.".format(vaccine))
for vaccine in eligible_with_appt:
    print("- {} vaccine with an appointment.".format(vaccine))
```

Open the file **TASK3.py**

Save the file as **TASK3_**<your name>_<class>_<index number>**.py**

10 Identify and correct the errors in the program so that it works according to the description given. [10]

Save your program.

Task 4

You have been asked to create a program for a simple calculator that performs the four basic mathematical operations, **+**, **-**, **×** and **÷**.

The program should allow you to:

- Enter a mathematical expression represented by two **positive real numbers** separated by a basic operator between them. The multiplication operator (**×**) is represented by `"*"` and the division operator (**÷**) is represented by `"/"`, for example, `"1.2*3.3"` and `"5/3"`

- Output the evaluated result, beginning with an equal sign (=), for example `"=3.96"`. You may use the function `format(float,'.10g')` to limit the result to 10 significant digits.

- Asks the user to input the next mathematical expression to be evaluated. This program should repeat until the user enters an empty input, i.e. `""`.

Example:

```
Enter an expression: 0.2+0.3
=0.5
Enter an expression: 9-4
=5
Enter an expression: 1.5*8
=12
Enter an expression: 12/4.0
=3
Enter an expression:
>>>
```

11 Write your program and test that it works. [5]

Save your program as **TASK4_Q11** _<your name>_<class>_<index number>**.py**

12 When your program is complete, use the following test data:

```
0.1+0.2
3-2
0.5*6
9/3.0
```

End the program with an **empty input**, i.e. `""`.

Take a screenshot of the outputs displayed on the screen and save as: [2]
TASK4_Q12 _<your name>_<class>_<index number>

Save your files in either **.png** or **.jpg** format.

13 Save your program as **TASK4_Q13** _<your name>_<class>_<index number>**.py**

Extend your program to check for any division by zero. If a division by zero is found in the expression, inform the user that the expression cannot be evaluated and the user will be required to key in a new expression. [3]

Save your program.

14 Extend your program to evaluate expressions with any number of operators, as described below:

(a) When there are **no operators** in the expression, the expression is simply returned together with an equal sign "=". [2]

For example, the expression "4.2" gives the result "=4.2".

Save your program as **TASK4_Q14a** _<your name>_<class>_<index no.>**.py**

(b) When there are **more than one operator** in the expression, the multiplication `"*"` and division `"/"` operators are evaluated first before the addition `"+"` and subtraction `"-"` operators are evaluated.

For example, the expression `"3+2*2"` gives the result `"=7"` as **2×2** is evaluated first before **3** is added.

When there are **consecutive** multiplication `"*"` and division `"/"` operators in the expression, the first operator on the left will evaluated first.

For example, the expression `"3/3*3"` gives the result `"=3"` as **3/3** is evaluated first before it is multiplied by **3**.

[8]

Save your program as **TASK4_Q14b** _<your name>_<class>_<index no.>**.py**

END OF PAPER

QUICK REFERENCE FOR PYTHON

1 Identifiers

When naming variables, functions and modules, the following rules must be observed:

- Names should begin with character ‘a’-‘z’ or ‘A’-‘Z’ or ‘_’ and followed by alphanumeric characters or ‘_’ .
- Reserved words should not be used.
- User-defined identifiers are case sensitive.

2 Comments and Documentation Strings

```
# This is a comment

"""
    This is a documentation string over multiple
    lines
"""
```

3 Input/Output

```
print (“This is a string”)

s = input (“Instructions to prompt for data entry.”)
```

4 Import

import <module>

e.g. **import** math

5 Data Type

Data Type	Notes
int	integer
float	real number
bool	boolean
str	string (immutable)
list	Series of values

6 Assignment

Assignment Statement	Notes
a = 1	integer
b = c	variable
d = “this is a string”	string
mylist = [1, 2, 3, 4,5]	list or array

7 Arithmetic Operators

Operator	Notes
+ -	plus, subtract
* /	multiply, divide
%	remainder or modulus
**	exponential or power
//	quotient of the floor division

8 Relational Operators

Operator	Notes
==	equality
!=	not equal to
> >=	greater than, greater than or equal to
< <=	less than, less than or equal to

9 Boolean Expression

Boolean Expression	Notes
a and b	logical and
a or b	logical or
not a	logical not

10 Iteration

while loop	for loop
while conditions(s): <statement(s)>	**for** i **in** range(n): <statement(s)> **for** record **in** records: <statement(s)>

11 Selection

Type 1	Type 2	Type 3
if condition(s): <statement(s)>	**if** condition(s): <statement(s)> **else:** <statement(s)>	**if** condition(s): <statement(s)> **elif** condition(s): <statement(s)> **else:** <statement(s)>

12 Built-in Functions

(a) Basic Functions

abs( )	chr( )	float( )	input( )	int( )
ord( )	print( )	range( )	round( )	str( )
format( )				

(b) Mathematical Functions

ceil( )	exp( )	fabs( )	floor( )	log( )
max( )	min( )	pow( )	sqrt( )	trunc( )

(c) String Functions

endswith( )	find( )	isalnum( )	isalpha( )	isdigit( )
islower( )	isspace( )	isupper( )	len( )	lower( )
startswith( )	upper( )			

13 Reserved Words

Reserved words cannot be used as identifiers. They are part of the syntax of the language.

False	None	True	and	as
assert	break	class	continue	def
del	elif	else	except	finally
for	from	global	if	import
in	is	lambda	nonlocal	not try
or	path	raise	return	
while	with	yield		