ARITHMETIC

2020 CODING HIGH SCHOOL
PROBLEM 1

For this exercise, you must write a program that asks for two numbers and an operator, and then performs the given arithmetic operation on those numbers.

Your program must prompt for input as shown in the examples below. For each of the first two prompts, your program will be given an integer in decimal (base 10) form, followed by the enter key. The final input will consist of one of the four following characters: + - * /, followed by the enter key. This character will indicate the arithmetic operation to perform with the previously given numbers; + for addition, - for subtraction, * for multiplication, and / for division. You may safely assume that the integers you are given will be between -2,147,483,648 and 2,147,483,647 (the minimum and maximum values of a signed 32-bit integer). You may also safely assume that the integer input will consist only of digits 0 through 9 and that no separators (such as commas) will be used. If division by zero is attempted, your program must output "Error: Division by Zero". Your program's output must not include any separators (such as commas).

*Your program's output must match the examples given below **exactly**. Note carefully the spelling, capitalization, punctuation, and spacing of the input prompts and output. The input that will be given to your program is highlighted in **red**.*

EXAMPLE RUN 1

```
1st Number: 10
2nd Number: 10
Operator: *
10 * 10 = 100
```

EXAMPLE RUN 2

```
1st Number: 10
2nd Number: 0
Operator: /
Error: Division by Zero
```

ACCOUNT BALANCE

2020 CODING HIGH SCHOOL
PROBLEM 2

For this exercise, you must write a program that prompts for a series of transaction amounts and calculates the remaining account balance. Your program must warn the user if the account balance is low or negative.

Your program must prompt for input as shown in the examples below. For each prompt, your program will be given a number with no more than two digits after the decimal, followed by the enter key. Positive numbers represent deposits to the account and negative numbers represent withdrawls. After each number is input, your program must calculate the remaining balance in the account. If the balance is less than $250, your program must output a warning which contains the remaining balance, as shown in the examples below. If the balance is less than $0, your program must output an alert about the account being negative, as shown in the examples below. If the user enters a transaction amount of 0, your program must output the final balance and exit.

You may safely assume that the numbers you are given will be between -1,000,000 and 1,000,000. You may also safely assume that the number input will consist only of digits 0 through 9 and optionally a decimal point; no separators (such as commas) will be used. Your program's output must not include any separators (such as commas). Whenever your program outputs the balance, it must be prefixed with a dollar sign ($) and have exactly two digits after the decimal.

Your program's output must match the examples given below ***exactly****. Note carefully the spelling, capitalization, punctuation, and spacing of the input prompts and output. The input that will be given to your program is highlighted in* ***red****.*

EXAMPLE RUN 1

```
Transaction: 1000
Transaction: -456
Transaction: -321
WARNING: Your balance of $223.00 is less than $250.
Transaction: -275
ALERT: Your account has a negative balance of $-52.00!
Transaction: 0
Your final balance is $-52.00.
```

EXAMPLE RUN 2

```
Transaction: 1000
Transaction: -12.34
Transaction: -105.67
Transaction: -546.78
Transaction: -43.21
Transaction: -321.98
ALERT: Your account has a negative balance of $-29.98!
Transaction: 1000
Transaction: 0
Your final balance is $970.02.
```

VOWELS

2020 CODING HIGH SCHOOL PROBLEM 3

For this exercise, you must write a program that prompts for a string of text input and finds all of the vowels in that input. Your program must then output the total number of vowels, and then indicate which, if any, vowels were not present in the input.

Your program must prompt for input as shown in the examples below. Your program will be given one or more English sentences as input, followed by the enter key. Your program must then count the total number of vowels in the input. For the purposes of this exercise, vowels will consist only of the letters “a”, “e”, “i”, “o”, and “u”; “y” will not be considered a vowel. Finally, your program must indicate any vowels which were not present in the input; if all vowels were represented, no additional lines should be output.

You may safely assume that the input will consist of no more than 1,000 characters, that all characters will be encoded as valid UTF-8, and that your program will be given the locale `en_US.utf8`. You will not need to account for characters with diacritical marks (e.g. á, ü, ñ, etc.).

Your program's output must match the examples given below ***exactly****. Note carefully the spelling, capitalization, punctuation, and spacing of the input prompts and output. The input that will be given to your program is highlighted in* ***red***.

EXAMPLE RUN 1

```
Message: A brown wood chair, from oak and ash, is boring; think of painting it black and
indigo.
Number of vowels: 25
Vowel(s) not present: e u
```

EXAMPLE RUN 2

```
Message: "Declarations of a fixed opinion, and of determined resolution never to change
it, neither enlighten nor convince us." --Benjamin Franklin
Number of vowels: 46
```

ATTENDANCE

2020 CODING HIGH SCHOOL
PROBLEM 4

For this exercise, you must write a program that determines which members of a TSA chapter attend the weekly chapter meetings the most often.

Your program must first prompt for the number of weeks for which attendance data will be given. For each week, your program must then prompt for the names of the students who attended. Only first names will be given, and each name will be separated with a space. After all of the attendance data has been input, your program must then output the name of the student or students with the most frequent attendance. After the names of the best attending students, your program must also output the number of meetings those students attended.

You may safely assume that each line of input will consist of no more than 1,000 characters, that all characters will be encoded as valid UTF-8, and that your program will be given the locale `en_US.utf8`. You will not need to account for characters with diacritical marks (e.g. á, ü, ñ, etc.). You may also safely assume that each student's first name will be unique. The names will be given in no particular order.

*Your program's output must match the examples given below **exactly**. Note carefully the spelling, capitalization, punctuation, and spacing of the input prompts and output. The input that will be given to your program is highlighted in **red**.*

EXAMPLE RUN 1

```
Number of Weeks: 3
Week 1 Attendance: Emma Ava Liam Oliver
Week 2 Attendance: Amelia Emma Lucas Liam
Week 3 Attendance: Emma Amelia Benjamin Lucas
Best Attendance: Emma, 3 meetings
```

EXAMPLE RUN 2

```
Number of Weeks: 4
Week 1 Attendance: Emma Ava Liam Oliver
Week 2 Attendance: Amelia Emma Lucas Liam
Week 3 Attendance: Emma Amelia Benjamin Lucas
Week 4 Attendance: Ava Sophia Liam Mason
Best Attendance: Emma Liam, 3 meetings
```

AVERAGE TAX RATE

2020 CODING HIGH SCHOOL
PROBLEM 5

For this exercise, you must write a program that calculates the average tax rate for a given set of tax brackets and amount of income.

With a bracketed tax system, each dollar of income is allocated to one of the brackets, starting at the lowest bracket until it is filled, then moving up the brackets until all income has been allocated. Each bracket is then taxed at a different rate. To illustrate, let's walk through the process of calculating the total tax on $100 of income with a highly simplified set of income tax brackets:

Income Bracket	Bracket Tax Rate	Allocation of $100	Tax
$1 - $10	10%	$ 10	$ 1
$11 - $30	20%	$ 20	$ 4
$31 - $60	30%	$ 30	$ 9
$61 - $90	40%	$ 30	$ 12
$91+	50%	$ 10	$ 5
	Total	$ 100	$ 31

As you can see, the first $10 of the $100 income was allocated to the first bracket, where it was taxed at 10% for a tax amount of $1. The second bracket covers dollars 11 though 30 (inclusive), a range of $20. Accordingly, $20 of the $100 income is allocated to the second bracket and taxed at 20% for a tax amount of $4. After two brackets, $30 of the $100 income have been taxed, with the remaining $70 yet to be taxed. The third bracket covers dollars 31 through 60 (inclusive), a range of $30. Accordingly, $30 of the $100 income is allocated to the third bracket and taxed at 30% for a tax amount of $9. The fourth bracket covers dollars 61 through 90, a range of $30. Accordingly, $30 of the $100 income is allocated to the fourth bracket and taxed at 40% for a tax amount of $12. After four brackets, $90 of the $100 income have been taxed, with only $10 remaining to be taxed. The fifth and final bracket covers all amounts $91 or greater but there is only $10 left to taxed in our example. Those $10, therefore, are allocated to the fifth bracket and taxed at 50% for a tax amount of $5. Now all $100 have been taxed. To calculate the total tax amount, we simply add the tax amounts from each bracket: 1 + 4 + 9 + 12 + 5 = $31. The final step is to calculate the average tax rate for the full $100. To do so, we simply divide the total tax amount, $31, by the total income, $100: 31 ÷ 100 = 0.31 or 31%.

Your program must first prompt for the number of brackets to be used in the tax calculation. Then, for each bracket, it must prompt for the maximum value of the bracket, followed by the tax rate for that bracket. On the last bracket however, you should assume that it covers all remaining income levels and only prompt for the tax rate. After collecting all of the bracket data, it must then prompt for the total income to be taxed. It must then output, for each bracket, the share of the income allocated to that bracket, and the tax amount. It must then output the total tax amount. Finally, it must output the average tax rate as a percentage.

You may safely assume that all input will consist of non-negative integers less than 1,000,000,000 (one billion). Tax amounts must be output with exactly two digits after the decimal, the average tax rate must be output with one digit after the decimal, and all other amounts should be displayed as whole integers. Values representing dollar amounts must be preceded by a dollar sign ($) and values representing percentages must be followed by a percent sign (%). All numbers output by your program must use a comma as a thousands separator, if applicable.

Your program's output must match the examples given below ***exactly****. Note carefully the spelling, capitalization, punctuation, and spacing of the input prompts and output. The input that will be given to your program is highlighted in* ***red****.*