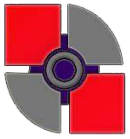
UTM
UNIVERSITI TEKNOLOGI MALAYSIA



PERSAKA
PERSATUAN MAHASISWA SAINS KOMPUTER

PROGRAMMING BATTLE

INDIVIDUAL QUESTIONS

DATE OF BATTLE

10th DECEMBER 2020

INDIVIDUAL

1. Write a program that determines which of **six states** within Malaysia (Johor, Selangor, Perak, Kedah, Pahang, Melaka) had the **fewest** and **highest** reported automobile accidents last year. **Tabulate** the input (number of accidents) received with title AUTOMOBILE ACCIDENTS 2019.

   Input validation: Do not accept negative numbers for accident number.

   Sample output:

```
Please enter number of accidents for Johor : 250
Please enter number of accidents for Selangor : 780
Please enter number of accidents for Perak : 400
Please enter number of accidents for Kedah : 600
Please enter number of accidents for Pahang : 530
Please enter number of accidents for Melaka : 850

-----------------------------------------------------
              AUTOMOBILE ACCIDENTS 2019
-----------------------------------------------------
              States         No Of Accidents
-----------------------------------------------------
               Johor                 250
            Selangor                 780
               Perak                 400
               Kedah                 600
              Pahang                 530
              Melaka                 850
-----------------------------------------------------


Melaka had the highest accidents with 850.
Perak had the lowest accidents with 400.
```

2. Have you heard of Pig Latin? It is a secret language formed from English by transferring the initial consonant or consonant cluster of each word to the end of the word and adding a vocalic syllable, usually “-ay”. In this question, you need to write a program that reads an array of words as input and converts each word to “Pig Latin.” To convert a word to Pig Latin you remove the first letter and place that letter at the end of the word. Then you **append the string “ay”** to the word except for the word that begins with **“l”, “m”, “n” and “p”**. For words that begin with “l”, “m”, “n” and “p”, the **string “ais”** should be appended. Tabulate the data with title PIG LATIN CONVERSION.

   Sample output:

```
Enter a string: Compfair is fun

Enter a string: Programming battle

Enter a string: Stay safe

Enter a string: Win the prize

Enter a string: UTM Awesome

------------------------------------------------------------------------
                    PIG LATIN CONVERSION
------------------------------------------------------------------------
 Before Convertion                    After Conversation
------------------------------------------------------------------------
   Compfair is fun                       ompfair is funCay
Programming battle                   rogramming battlePais
         Stay safe                             tay safeSay
     Win the prize                         in the prizeWay
       UTM Awesome                           TM AwesomeUay
------------------------------------------------------------------------
```

3. Write a program to print a Pascal’s Triangle by asking the user to input a value for row number. After printing the triangle, you have to compute the value of a given position in Pascal's Triangle. The way to calculate any given position's value is to sum up the numbers to the position's right and left in the preceding row. For example, to compute the middle number in the third row, you add 1 and 1; the sides of the triangle are always 1 because you only add the number to the upper left or the upper right (there being no second number on the other side). The program should prompt the user to input a row and a position in the row. The program should ensure that the input is valid before computing a value for the position.

   Sample output:

```
Please input a row number: 10

                        1
                        1   1
                        1   2   1
                        1   3   3   1
                        1   4   6   4   1
                        1   5   10   10   5   1
                        1   6   15   20   15   6   1
                        1   7   21   35   35   21   7   1
                        1   8   28   56   70   56   28   8   1
                        1   9   36   84   126   126   84   36   9   1

 Please input a row and a position along the row: 10 6

 Value at row 10 and position 6 is 126
```

4. We have learnt that the position of a digit in an integer indicates the weight of that digit towards the value of the integer number. For instance, in the base 10 number 253 we know that 3 has the weight 10^0, 5 has the weight 10^1, and 2 has the weight 10^2, yielding the value 3 x 10^2 + 6 x 10^1 + 2 x 10^0, or just 200 + 50 + 3. The same mechanism is used for numbers expressed in other bases. Most people presume that the numbers they encounter everyday are expressed using base 10, but we know that other bases are possible. In particular, the number 253 in base 9 or base 14 represents a totally different value than 253 in base 10.

   For this problem your program will be presented with a sequence of pairs of integers. Let's call the members of a pair *P* and *Q*. What your program is to do is determine the smallest base for *P* and the smallest base for *Q* (likely different from that for *P*) so that *P* and *Q* represent the same value.

   Consider, for example, the integers 12 and 6. Certainly these are not equal if base 10 is used for each. But suppose 12 was a base 4 number and 6 was a base 7 number? 12 base 4 = 1 x 4^1 + 2 x 4^0, and 6 base 7 is equal to 6. So 12 and 6 can be equal, if you select the right bases for each of them.

   Sample Output:

```
12 6
12 (base 4) = 6 (base 7)
10 B
10 (base 11) = B (base 12)
2 1
2 is not equal to 1 in any base 2..36
567 789
567 is not equal to 789 in any base 2..36
34 15
34 (base 5) = 15 (base 14)
```

5. You have to create a program to simulate a real life blackjack game. In the program, you should create a function that takes an array of card numbers and verifies if the sum of the cards' value exceeds 21. If the sum exceeds 21, return true and if the sum is less than or equal to 21, return false.

   Values of the cards are 2 - 10. The face cards, which are J, Q and K are counted as 10. In addition, aces could be counted either as 1 or 11, it depends on which value is the best for the player at the moment. For instance, if the ace value of 11 can contribute to win while value of 1 contributes to lose, then go with 11.

   Sample output:

```
How many cards do you have?  3


Enter the your #1 card value: 6
Enter the your #2 card value: 3
Enter the your #3 card value: 9


exceedTwentyOne([ 6  3  9 ]) -> False
```

```
How many cards do you have?  3


Enter the your #1 card value: J
Enter the your #2 card value: Q
Enter the your #3 card value: 3


exceedTwentyOne([ J  Q  3 ]) -> True
```

```
How many cards do you have?  4


Enter the your #1 card value: 5
Enter the your #2 card value: 6
Enter the your #3 card value: J
Enter the your #4 card value: 8


exceedTwentyOne([ 5  6  J  8 ]) -> True
```

```
How many cards do you have?  2


Enter the your #1 card value: J
Enter the your #2 card value: Q


exceedTwentyOne([ J  Q ]) -> False
```