Group A4 – PA1's Question 17/18 - Not included!

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q7 [A1 –1]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.
2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

Develop a Python program that takes a sequence of integers as input and outputs the number of powers of 2 in that sequence. A power of 2 is 2^n where n is a positive integer.

Example execution session:

```
Input: 23 32 1 45 67 128 2048 1024 87 16
Number of powers of two: 6
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q8 [A1 –2]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.
2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

Develop a Python program to take as input 3 words and to check whether the first two words exist within the third word. If either of the first 2 words exists in the third word, not as a complete word, but the letters of each word is in order but scrambled within the third word, then output should be given as SHUFFLE. Otherwise if both of the first two words exists in the third word, as the complete word itself, then out should be given as NOT A SHUFFLE.

Example execution session 1:

```
Enter word 1: joy
Enter word 2: enable
Enter word 3: ksjdotiyejnshaqwbolpe
Output: SHUFFLE
```

Example execution session 2:

```
Enter word 1: joy
Enter word 2: enable
Enter word 3: pijoydkfenables
Output: NOT A SHUFFLE
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q23 [A2 –1]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.
2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

There are *p* players in a tennis tournament in which two players play a tennis match against each other and one will win. The matches are played over several *rounds,* which are numbered 1, 2, 3 ….. Initially, all *p* players enter round 1. The rounds will proceed as follows.

- Let *i* the current round, let *n* be the number of the players entering round *I* (*n* = *p* when *i*=1)
- Let $m = 2^k$ for some integer $k>0$, and *m* be the largest possible such that $m<=n$.
- *m* players will play m/2 matches in round *I*; the *m*/2 winners will enter round *i*+1; the *m*/2 losers will not play any more matches (they will not enter round *i*=1)
- (*n* - *m*) players who enter round *I* will not play any match in round *I* and enter round *i*+1 directly

- The tournament ends after the round in which *n*=2; there is only 1 match (the final match) and the winner in that is the winner of the tournament

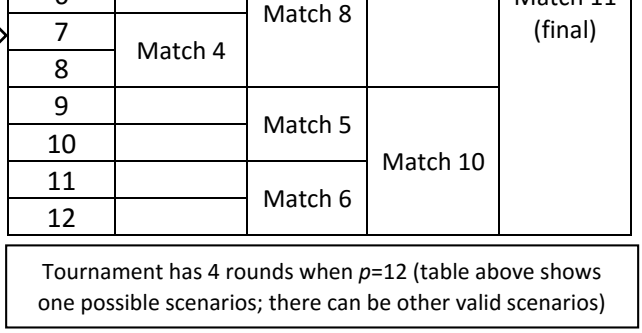
Develop a Python program that takes as input *p*, where *p* (2<=*p*<=1000) is the number of players entering the tournament, and outputs the number of rounds and the total number of matches played until a winner is selected. You may handle unexpected inputs appropriately.

Example execution session 1:

```
Number of players, p: 12

Number of rounds=4, matches=11
```

Explanation for case *p*=12



Tournament has 4 rounds when *p*=12 (table above shows one possible scenarios; there can be other valid scenarios)

Example execution session 2:

```
Number of players, p: 29

Number of rounds=7, matches=28
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q24 [A2 –2]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.
2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

To identify companies whose stocks are worth buying, Kamal, a stock broker has come up with a new idea. If a company on a specific day has a stock price which is greater than (10 + the average stock price

of that week), such days are called “valued days”. If a company has 3 valued days in a week, Kamal recommends buying stocks of that company. Otherwise he recommends not to buy.

Example:
Suppose the daily stock prices of a company over a week are as follows:

```
70.5 78.9 90.7 33.2 56.0 40.2 22.1
```

The average stock price for the week is 55.9. There are 3 “valued days” (days 1, 2 and 3) on which the stock price is greater than (10 + average). Therefor Kamal will recommend to buy.

Develop a Python program that takes as input the daily stock prices over a week separated by space and outputs the number of “valued days” and the recommendation whether to buy or not to buy stocks (output as either “RECOMMENDED” or “NOT RECOMMENDED”). Each input (daily stock price) is a positive number, possible a non-integer. You may handle unexpected inputs appropriately.

Example execution session 1:

```
Input stock prices of the week: 70.5 78.9 90.7 33.2 56.0 40.2 22.1
Number of valued days: 3
RECOMMENDED
```

Example execution session 2:

```
Input stock prices of the week: 33.5 46.9 40.7 43.2 26.0 30.2 32.1
Number of valued days: 1
NOT RECOMMENDED
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q19 [A3 –1]

Instructions:

1. Name your source file as “p1<StudentNum>.py” and save it in a folder/directory named “p1” in your home directory.
2. Upload the source file, “p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **“150001C”** then the name of your source file will be **“p1-150001C.py”**

The Problem

A *garland* word is a word that starts and ends with the same N letters in the same order, for some N where 0 < N < (length of the word). The maximum N that maintains this property is the *garland word's degree.* For example, "onion" is a garland word of degree 2, because the first two letters "on" are the same as its last letters. The name "garland word" comes from the fact that you can make chains of words as follows.

```
onionionionionionionionionionionionionionionion...
```

Develop a Python program that takes as input a word and outputs the degree of the word if it is a garland word. And 0 otherwise. An input word can have only alphabetical characters. Lower-case- Upper case differences between letters must be ignored; that is, for example, "a" is considered the same as "A". You may handle unexpected inputs appropriately. You can use the following functions if needed.

Function	Description	Example
`max()`	Returns the larger between two arguments	`max(3,5)` returns `5`
`isalpha()`	Return if the character is alphabetic	`'c'.isalpha()` returns `True` `'%'.isaplha()` returns `False`
`lower()`	Converts a character to lowercase	`'C'.lower()` returns `'c'`
`upper()`	Converts a character to uppercase	`'a'.upper()` returns `'A'`

Example execution sessions are shown below:

```
Enter the input word: PrograMmer
Degree = 0

Enter the input word: CeraMIc
Degree = 1

Enter the input word: alfalfa
Degree = 4

Enter the input word: onion
Degree = 2
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q20 [A3 –2]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.

2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

You are going to balance a word at one of its letters called the *balance point.* The parts of the word that are on the left and right of the balance point are *the left-part and right-part*, respectively. A word can be balanced if a balance point exists such that the weights of left and right parts are equal. But some words cannot be balanced. The weight of a part of the word is the sum of the weights of its letters. The weight of a letter = (its position in the English alphabet) x (its distance from the balance point). The positions are A=1, B=2, C=3, ….,, Z=26 (we ignore whether it is lower or upper case).

Example: Consider the word "STEaD". This word balances at "T", as shown below.
Left-part = "S"; weight of left-part = weight of "S" = position in alphabet x distance = 10 x1 = 19
Right-part = "EaD"; weight of right-part = weight of "E" + weight of "a" + weight of "D"
= (5 x 1) + (1 x 2) + (4 x 3) = 19
Therefore T is the balance point and that the weight of each part is 19.

Develop a python program that takes as input a word, determines if it can be balanced, and, id yes, then outputs the balance point and the weight of one part, otherwise outputs "Cannot be balanced". Input must have only letters (alphabetical characters) and can be either lower or upper case. You may handle unexpected inputs appropriately. You can use the following function if needed.

Function	Description	Example
`isalpha()`	Return if the character is alphabetic	`'c'.isalpha()` returns `True` `'%'.isaplha()` returns `False`
`lower()`	Converts a character to lowercase	`'C'.lower()` returns `'c'`
`upper()`	Converts a character to uppercase	`'a'.upper()` returns `'A'`

Example execution sessions are shown below:

```
Enter the input word: ConsubsTantiation
Balance point letter: "a", weight=456

Enter the input word: Superglue
Cannot be balanced!
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016

Programming Assignment 1
Time: 1 hour only
Q1 [B1 – 1]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.
2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

Swedish children have developed a secret language that only kids know, so they can hide secrets from their confused parents. This language is known as "Rövarspråket" (means "Robber's language").

This secret language is not very complicated, each ordinary English word is code by replacing each consonant with the consonant doubled (repeated) and with an "o" in between. If the consonant is in uppercase, the repeating consonant is in lowercase. So the consonant "b" is replaced "bob", "r" is replaced by "ror", "n" is replaced with "non", "G" is replaced by "Gog" and so on. Vowels and other non-alphabetic characters are left intact (not changed).

Example:

Input:	Hello World!
Output (Encoded):	Hohelollolo Wowororloldod!
Input:	b
Output (Encoded):	bob!

Develop a Python program to encode a given input sentence into "Rövarspråket" and outputs the encoded string. You may handle unexpected errors appropriately.

You may use the `isaplha()` Python function to check whether a given character is alphabetic.
Examples: `"C".isalpha() => True` `"%".isalpha() => False`

You may use the `lower()` Python function to convert a character to lowercase and `upper()` Python function to convert a character to uppercase.
Examples: `"C".lower() => 'c'` `"a".upper() => 'A'`

Example execution session:

```
Input: Hello World!
Output: Hohelollolo Wowororloldod!

Input: where is god!
```

```
Output: wowhoherore isos gogodod!
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q2 [B1 – 2]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.
2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

A *palindromic number* (or *palindrome*) is a number that remains the same when its digits are reversed (that is, it reads the same from left-to-right and right-to-left)
Examples: 1, 2, ….., 9, 11, 22, ……, 99, 111, …… 484, 676, 10201, 12321

A double-base palindrome is a palindromic number that maintains palindromic property in two bases.
Examples: $99_{10} = 1100011_2$
$585_{10} = 1001001001_2$

Develop a python program that repeatedly takes an integer (1<=n<=100000) as input and outputs whether the given number is a **Double Base palindrome** in base 10 and base 2. You have to continue to take inputs and repeat this, until the user enters -1 as the input. You may handle unexpected inputs appropriately.

You may use the Python built-in function "`bin()`" in your program to obtain the binary (base 2) representation of a number. Example: `bin(23) = '0b10111'`

Sample execution of the program should be similar to the following.

```
Enter Input: 99
Yes
Enter Input: 585
Yes
Enter Input: 100
```

No
```
Enter Input: 1222
```
No
```
Enter Input: 56982
```
No

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q3 [B2 – 1]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.
2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

Develop a python program that takes as input a word (consisting only of letter from the alphabet) and output if the letters in the input word are in alphabetical order (IN ORDER), reverse alphabetical order (IN REVERSE ORDER) or neither (NOT IN ORDER). Each letter can be either lower or upper case but this must be ignored. You may use the following functions if you need.

Function	Description	Example
`ord()`	Returns the decimal value of the ASCII code	`ord('a')` returns `97`
`isalpha()`	Return if the character is alphabetic	`'c'.isalpha()` returns `True` `'%'.isaplha()` returns `False`
`lower()`	Converts a character to lowercase	`'C'.lower()` returns `'c'`
`upper()`	Converts a character to uppercase	`'a'.upper()` returns `'A'`

Example execution session 1:
```
Enter word: aaaaa
Output: aaaaa IN ORDER
```
Example execution session 2:
```
Enter word: alMost
Output: alMost IN ORDER
```
Example execution session 3:
```
Enter word: WrongeD
```

```
Output: WrongeD IN REVERSE ORDER
```

Example execution session 4:

```
Enter word: random
Output: random NOT IN ORDER
```

Example execution session 5:

```
Enter word: toFfeE
Output: toFfeE IN REVERSE ORDER
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q4 [B2 – 2]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.
2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

"Disemvoweling" means removing the vowels and spaces between words from a given text. The letters a,e,I,o,u are vowels. The idea is to make the text difficult but not impossible to read. We will keep the vowels we removed (in their original order) as well.
Example:

Original string (INPUT): `two drums and a cymbal fall off a cliff`
Disemvoweled string: `twdrmsndcymblfllffclff`
Removed vowels: `ouaaaaoai` (In original order)

Develop a python program that takes as input a string of words separated by space, disemvowel it and print disemvoweled string and removed vowels (in order of appearance in the input). Input can contain only alphabetic characters (upper and lower case) and space. You have to maintain the case of each letter in the output.

Example execution session 1:

```
Enter the input: Hello WorlD
Disemvoweled: HllWrlD
Vowels: eoo
```

Example execution session 2:

```
Enter the input: two drums and a cymbal fall off a cliff
Disemvoweled: twdrmsndcymblfllffclff
Vowels: ouaaaaoai
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q5 [B3 – 1]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.
2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

A gaming company tracks the number of games sold, when a customer buys a game, the number of games sold is increased by one. At that point, if the number of games sold, **is not** divided by any number between 2 and 10, the developer gets a reward.

Develop a python program that takes an integer n, ($1<=n<=10^4$), and output the number of rewards.

When inputted 29, the rewards are given at 1, 11, 13,17,19,23 and 29. So the number of rewards are 7.

Example execution 1:

```
Enter Input: 29
Number of rewards: 7
```

Example execution 2:

```
Enter Input: 12
Number of rewards: 2
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q6 [B3 –2]

Instructions:

1. Name your source file as “p1<StudentNum>.py” and save it in a folder/directory named “p1” in your home directory.
2. Upload the source file, “p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **“150001C”** then the name of your source file will be **“p1-150001C.py”**

The Problem

Develop a Python program which takes any positive integer n; (10 < n < 10,000) as an input and print 0 with n if the number divisible by 3. If input n is not divide by 3, then print 1 with n if the consecutive integer n+1 is divisible by 3 or print -1 with n if the consecutive integer n-1 is divisible by 3. This process is continued for the each quotient obtained after division by 3 until n=1 is obtained.

Example execution 1:

```
Enter Input: 998
Output:
998  -  1
333  -  0
111  -  0
 37  -  -1
 12  -  0
  4  -  -1
  1
```

Example execution 2:

```
Enter Input: 243
Output:
243  -  0
 81  -  0
 27  -  0
  9  -  0
  3  -  0
  1
```

Example execution 3:

```
Enter Input: 135
Output:
135  -  0
 45  -  0
 15  -  0
  5  -  1
  2  -  1
  1
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q13 [B4 –1]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.
2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

Jimith entered into a calculator the final answer for a homework problem, left the calculator on the table and went to sleep. Meanwhile his little brother, Arosh found the calculator and started to press keys randomly. Now the number displayed on the calculator is a string N of digits with two parts: the "prefix" (P) which is the substring of digits first entered by Jimith and the "suffix" (S), which is the substring of random digits later entered by Arosh. (i.e. N = PS). After waking up, Jimith got to know that Arosh had pressed at least one random key. Unfortunately Jimith has forgotten P, the number he himself entered: he only remembers that it was divisible by 4. Now he wants to find out P.

Develop a Python program to take as input N, the final string of digits on the calculator display, and output possible prefixes P which are divisible by $. A substring can start with zero. Note that the 3<= (length of N) <=16 and S has at least one digit. You may handle unexpected inputs appropriately.

Example: If N = "447295", then there are 3 possible prefixes P divisible by 4: 4472, 44 and 4.

Example execution session1:

```
Number displayed N: 124
Possible prefixes: 12
```

Example execution session 2:

```
Number displayed N: 447295
Possible prefixes: 4472, 44, 4
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016

Programming Assignment 1
Time: 1 hour only
Q14 [B4 –2]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.
2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

The life goes up and down, just like nice sequences. A sequence $t_1, t_2, ……., t_n$ is called a nice sequence following two conditions are satisfied:

- $t_i < t_{i+1}$ for each odd $i<n$
- $t_i > t_{i+1}$ for each even $i<n$

Otherwise, the sequence is not nice.

3 example nice sequences: (2, 8), (1, 5, 1) and (2, 5, 1, 100, 99, 120)
3 example not nice sequences: (1, 1), (1, 2, 3) and (2, 5, 3, 2)

Develop a Python program that takes as input a sequence of positive integers and outputs whether it is nice or not. Input integers are separated by space. A sequence has at least 2 integers and at most 9 integers. You may handle unexpected inputs appropriately.

Example execution session 1:

```
Enter the sequence: 2 5 1 100 99 120
Result: nice
```

Example execution session 2:

```
Enter the sequence: 2 5 3 2
Result: not nice
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q15 [C1 –1]

Instructions:

1. Name your source file as “p1<StudentNum>.py” and save it in a folder/directory named “p1” in your home directory.
2. Upload the source file, “p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **“150001C”** then the name of your source file will be **“p1-150001C.py”**

The Problem

In number theory, a deficient is defined as an integer n for which the sum of divisors < 2n. The value $2n$ - $\sum(divisors)$ is called as the number’s deficiency. In contrast, an abundant number can be defined as an integer n for which the sum of divisors > 2n. The value $\sum(divisors)$ - $2n$ is the numbers’ abundance.

Example 01: Consider 21. Its divisors are 1, 3, 7 and 21 and their sum is 32. Because 32 is less than 2x21, the number is deficient. Its deficiency value is 2x21-32 = 10.

Example 02: Consider 12, its divisors are 1,2,3,4,5,12 and their sum is 28. Because 28 is greater than 2x12, the number is abundant. Its abundant value is 28-2x12 = 4.

If number is neither deficient nor abundant, it is called as a perfect number whose sum of divisors = 2n.

Develop a python program that repeatedly takes an integer (1<=n<=100000) as input and outputs whether the given number is deficient and its deficiency, abundant and its abundance or whether a number is perfect.

Example execution session:

```
Input n: 12
12 is abundant by 4
Input n: 6
6 is perfect
Input n: 21
21 is deficient by 10
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q16 [C1 –2]

Instructions:

1. Name your source file as “p1<StudentNum>.py” and save it in a folder/directory named “p1” in your home directory.

2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

Develop a python program that repeatedly takes as input a non-negative integer *n*, which consists of at least 2 and at most 100 digits. Then it should do the following.

- If n is divisible by 8, then output is "Yes".
- If n is not divisible by 8, then try to obtain *m*, the largest number divisible by 8 by removing a minimum number of consecutive digits from *n* (but re-arranging digits is not allowed) and do one of the following:
    - If such a number *m* exists, then output "Yes" and the value of m.
    - If such a number *m* does not exist, then output "No".

Example execution session are shown below for n=3454, 96, 10, 111111, and 964:

```
Input: 3454
Yes m=344

Input: 96
Yes

Input: 10
No

Input: 111111
No

Input: 964
Yes m=96
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q11 [C2 –1]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.

2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

We can write the cube of any given integer n as the sum of n consecutive odd integers.

Examples: $1^3 = 1 = 1$
$2^3 = 8 = 3 + 5$
$3^3 = 27 = 7 + 9 + 11$
$5^3 = 125 = 21 + 23 + 25 + 27 + 29$

Develop a Python program to take an integer n as input (1<=n<=20) and output the set of n consecutive odd numbers whose summation is equal to n^3. The program should repeat this process with the next input, until the input is -1. You may handle unexpected inputs appropriately.

Example execution session:

```
Enter Input: 5
Odd Number Sequence: 21 + 23 + 25 + 27 + 29
Enter Input: 8
Odd Number Sequence: 57 + 59 + 61 + 63 + 65 + 67 + 69 + 71
Enter Input: -1
Bye!
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q12 [C2 –2]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.
2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

A palindromic word (or palindrome) is a string that remains the same when its letters are reversed (that is, it remains the same when read left-to-right or right-to-left)

Examples: `madam, civic, reviver, rotor, refer, level, noon.`

Develop a python program to take as input a word and output whether the input word is palindrome or not. If not the program should identify the longest palindromic substring (whose length is at least 9) in the input word. For example even though "afternoon" is not a palindrome "noon" is a palindrome. If no such substring is found, output "There are no palindromes". Then the program should repeat this by taking the next input word and continuing in the same manner. The program will stop when the input word is "bye".

Note that the input must be a single word. You may handle unexpected inputs appropriately.

Example execution session:

```
Enter word: reference
Longest Palindrome: refer
Enter word: level
Longest Palindrome: level
Enter word: mouse
There are no palindromes
Enter word: Bye
Bye!
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q21 [C3 –1]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.
2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

<u>The Problem</u>

Swedish children have developed a secret language that only kids know, so they can hide secrets from their confused parents. This language is known as "Rövarspråket" (means "Robber's language").

This secret language is not very complicated, each ordinary English word is code by replacing each consonant with the consonant doubled (repeated) and with an “o” in between. If the consonant is in uppercase, the repeating consonant is in lowercase. So the consonant “b” is replaced “bob”, “r” is replaced by “ror“, “n“ is replaced with “non“, “G“ is replaced by “Gog“ and so on. Vowels and other non-alphabetic characters are left intact (not changed).

Example:

Input:	Hello World!
Output (Encoded):	Hohelollolo Wowororloldod!

Input:	b
Output (Encoded):	`bob`!

Develop a Python program that Swedish parents can use to decode what their kids are saying. The input string can have a maximum length of 200. You may handle unexpected errors appropriately.

You may use the `isaplha()` Python function to check whether a given character is alphabetic.
Examples: `“C”.isalpha() => True` `“%”.isalpha() => False`

You may use the `lower()` Python function to convert a character to lowercase and `upper()` Python function to convert a character to uppercase.
Examples: `“C”.lower() => ‘c’` `“a”.upper() => ‘A’`

Example execution session:

```
Input: Hohelollolo Wowororloldod!
Output: Hello World!

Input: wowhoherore isos gogodod!
Output: where is god!
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q22 [C3 –2]

Instructions:

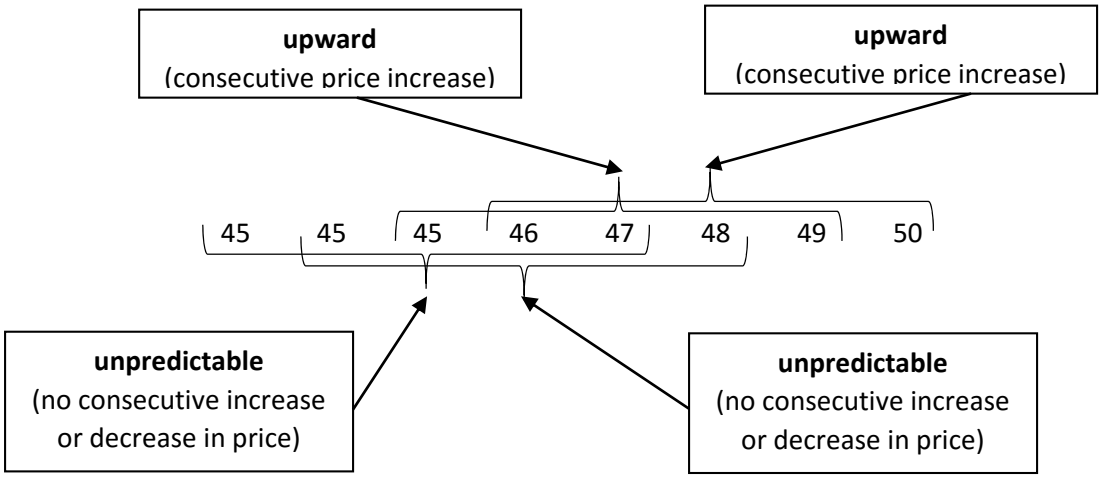
1. Name your source file as “p1<StudentNum>.py” and save it in a folder/directory named “p1” in your home directory.
2. Upload the source file, “p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

The Finance Ministry of the country Holland found, by research that the economic trend of the country can be roughly identified by the price of bread. If the regulated price of bread increases for 5 consecutive weeks, the economic trend is ***upward***. If the price decreases for five consecutive weeks, then the economic trend is ***downward***. If there is no such pattern in bread prices, the economic trend is ***unpredictable***.

Example: Price of bread over 8 weeks



In the above example, within the 8 week sequence, there are four 5 week windows. As can be seen the first two windows have an unpredictable trend and the last two windows have an upward trend.

Develop a python program that takes as input a sequence of weekly bread prices separated by space (5<=number of weeks<=20) and outputs the economic trend of each 5 week window. Each bread price is a positive integer. You may handle unexpected inputs appropriately.

Example execution session 1:

```
Bread Prices: 45 45 45 46 47 48 49 50
Trend: unpredictable, unpredictable, upward, upward
```

Example execution session 2:

```
Bread Prices: 45 44 43 42 41 44
Trend: downward, unpredictable
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q9 [C4 –1]

Instructions:

1. Name your source file as "p1<StudentNum>.py" and save it in a folder/directory named "p1" in your home directory.
2. Upload the source file, "p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **"150001C"** then the name of your source file will be **"p1-150001C.py"**

The Problem

Two strings are anagrams if they are written using the exact same letters, ignoring space, punctuation and capitalization. Each letter should have the same count in both strings.

For example Army and Mary are anagrams of each other.

Develop a Python program to take two strings as inputs and out whether they are anagrams. An input string can have multiple words separated by space. Only alphabetical characters are allowed in a word. You may handle unexpected inputs appropriately.

Example execution session 1:

```
Enter string one: Eleven plus two
Enter string two: Twelve plus one
These are anagrams
```

Example execution session 2:

```
Enter string one: Eleven plus four
Enter string two: Fourteen plus one
These are not anagrams
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]

CS1032 Programming Fundamentals
2015 Batch, Feb – June 2016
Programming Assignment 1
Time: 1 hour only
Q10 [C4 –2]

Instructions:

1. Name your source file as “p1<StudentNum>.py” and save it in a folder/directory named “p1” in your home directory.
2. Upload the source file, “p1<StudentNum>.py onto the relevant area on Moodle.
3. Write your name and other details in the above box and return this sheet to staff at the end.

Make sure you name your source file correctly. Example: If your student number is **“150001C”** then the name of your source file will be **“p1-150001C.py”**

The Problem

There are n number of students in a class. For the year end concert, the teacher wants to pair the students by their heights for a special dance. If the pair of students are of the same height, they can participate for the dance. If a student does not have a partner with matching height, he/she is assigned to the support crew. The teacher has a list of heights and wants to know how many pairs will be participating for the dance this year.

Develop a Python program to read a list of positive integers (the list of heights) separated by spaces as the input from the keyboard and output the number of dancing pairs form the class. Each integer n satisfies 100<=n<=200. You may handle unexpected inputs appropriately.

Example execution session 1:

```
Enter the heights: 135 130 135 150 140 135
Number of dancing pairs: 1
```

Example execution session 2:

```
Enter the heights: 135 130 145 135 140 145 150 155
Number of Dancing Pairs: 2
```

Example execution session 3:

```
Enter the heights: 135 130 135 135 140 135 135
Number of Dancing Pairs: 2
```

[Fill the top-right box on this page and return this sheet at the end to the staff. Failure to do so is considered as non-submission of the assignment. Do not write anything else unless asked by staff.]