Haskell Assignment
Programming and Paradigms, Dr. Pierpaolo Dondio

Due Date: Sunday 7th January 2024 (30% of total exam marks)

Define the following Haskell functions. Always include the function signature.

(Note: the marks displayed sum up to 100)

1. is_square. [4]

Define a function is_square that takes an integer (positive) and returns True if the number is the square of an integer number. DO NOT use the SQRT function.

2. freq_letter_pc [6]

Define a function **freq_letter_pc**, that displays the percentage of each letter instead of the number of occurrences (do not count symbols other than [a..z]). Transform the text in small case first. Example:

```
freq_letter_pc "this is text" =
[(t,0.3),(s,0.2),(i,0.2),(h,0.1),(e,0.1),(x,0.1)]
```

3. db in Haskell [10]

A small database contains two tables, one is a list of cities, identified by a city_id, city_name, city_population and country_ id. The other table is a list of countries (country_id, country_name). Table country is related by a 1 to many relation with table cities, country_id is the foreign key. The following is the content of the table:

```
cities
=[(1,"Paris",7000000,1),(2,"London",8000000,2),(1,"Rome",3000000,3),
(1,"Edinburgh",500000,2),(1,"Florence",50000,3),
(1,"Venice",200000,3), (1,"Lyon",1000000,1),(1,"Milan",3000000,3),
(1,"Madrid",6000000,4),
(1,"Barcelona",5000000,4),]

countries = [(1,"UK), (2,"France"), (3,"Italy"), (4,"Spain")]
```

Write the following Haskell functions:

a. get_city_above n – to get all the names of the cities with a population above n (=input)

```
get_city_above 6000000 = ["Paris","London","Madrid"]
```

b. get_city country_name – to get all the cities given a country_name (not country_ id! You need to have a function that, given the country_name gets the country_id)

```
get_city "Italy" = ["Rome","Milan","Florence","Venice"]
```

c. num_city– to list all the country and for each country the number of cities in each country.

```
num_city = [("UK",2),("Italy",4), ("Fance",2),("Spain",2)]
```

4. Euclidean distance between two lists [5]

Create a function eucl_dist to compute the euclidean distance between two vectors. Each vector is represented by a list of float number of unknown length. The two lists must have the same number of elements.

The distance between list x and y is defined as follows:

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + .. + (x_n - y_n)^2}$$

You can use the built-in Haskell function sqrt.

5. Language Identification [15]

The file lang.hs contains two lists of 26 float numbers each. The lists represent the frequency of each letter in English (list eng_freq) and Portuguese (pt_freq). For instance, the letter "b", the second letter in the alphabet, has a frequency of 1.492% in English and 1.04% in Portuguese.

Write a Haskell function get_lang that gets a text and print the message "The text is in English" if the text is written in English or "The text is in Portuguese" if the text is written in Portuguese (as a simplification, we only use text containing the 26 basic letters, without accents or other phonetic symbols). In order to identify the language, compute the frequency distribution of the letters in the text, store it into a list and check if the distribution is closer to eng_freq or pt_freq. You can use the function defined at exercise 2 and 6. Transform the text in small case first.

Using the function readfile, create a version of the program running from command line that takes a file as an input (command line parameter) and detect the language of the file.

6. Caesar Cipher [5]

A (simplified) Caesar cypher is one of the oldest and simplest forms of encryption. Given a positive integer number n (called the index) and a string of text (for simplicity, we only work with sentences containing small letters and the space symbol), each letter is shifted by n.
If, for instance, n=3, then “a” becomes “d”, “b” -> “e” , “c” -> “f” and so on…
It is circular, therefore “x” becomes “a”, “y” -> “b” and “z” -> “c”. The space character is unchanged.
The file encrypt.hs is the Haskell function (attached to this assignment) to read a file and encrypt it. The function has two inputs: the name of the file and the index (number used to shift the letters). All the text is converted to lower case before doing the encryption using the function toLower.
You are required to write a function c_decrypt that takes a file encrypted with a Caesar cypher, an integer positive number (the index of the cypher) and generate the decrypted file.

7. Statistical decoder [17]

Suppose that we receive an encrypted message, and we do not know the value of the index *n* used to generate the message. We just know that the message was written in English.

We need to identify the value of the index. Since we know that the original message was in English, we can identify the index by using a dictionary of English words. The idea is to try to decrypt the messages with all the 26 possible indexes, and select the one that generates a message with the highest number of English words.

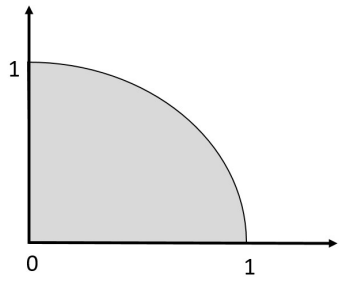
Part 1. Building an English Dictionary
You do not have an English dictionary available, but you have available the text of 3 famous English novels: Pride and Prejudice (J. Austen), Ulysses (J. Joyce) and Dorian Gray's Picture (O. Wilde). The novels are stored in the files pride.txt, Ulysses.txt and dorian.txt

You are required to write an Haskell function to read the three files and save into a single file dict.txt all the distinct unique words in the novels.

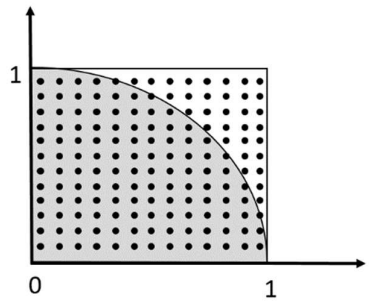
Part 2. Identifying the index
After having created the English dictionary, you are required to write an Haskell function guess_index that reads an encrypted file and identify the index of the cipher using the dictionary built in part 1 and save the decrypted message. The function choses the index that maximises the number of English words in the decrypted message. You can test your function with the sample files dit.txt.chp and italy.txt.chp

8. Approximating the area of a circle [7]

You are required to write an Haskell function to compute the value of the area in grey (a quarter of a circle).



You have to approximate the value of the area using a simple Montecarlo simulation. The idea is to divide the square of side 1 into many points and count the proportion of points that are inside the grey area, as depicted in the picture below.



HIGHER ORDER FUNCTIONS

9. Computing numerical series [15]

Write a higher order function `math_series` with two inputs: a function expressing a mathematical series (a function from a natural number to a float) and an integer positive number n,.
The function returns the sum of the first n elements of the series.
For simplicity, define a series from Float to Float.

For instance, if we have the series $\frac{1}{2^k}$ (that is $1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}$, ….) defined by the following function:

```
sample_series :: Float -> Float
sample_series k = 1 / (2^k)
```

then

```
math_series sample_series 4
```

should return 15/8=1.875. that is the sum of the first 4 elements of the series $\frac{1}{2^k}$ that is:

$$1+\frac{1}{2}+\frac{1}{4}+\frac{1}{8}=\frac{15}{8}$$

Using your function, shows that the sum of the series

$$\sum_{k=1}^{\infty}(-1)^{k+1}\left(\frac{4}{2k-1}\right).$$

goes to π when *k* goes to infinity, by showing that the greater is k, the closer the sum of the series gets to π.

10. Integral of a function [16]

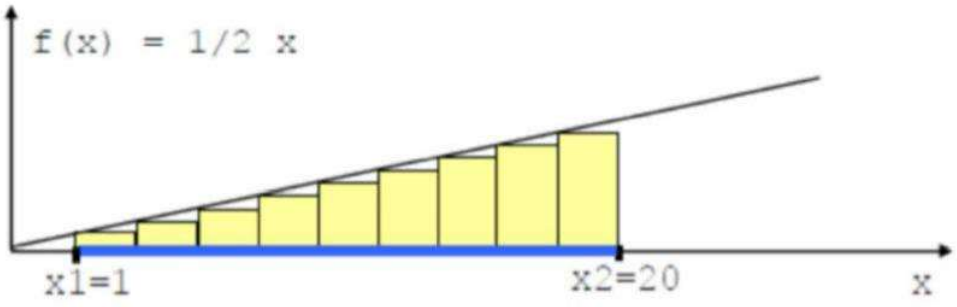
Define a function integral, that approximates the integral of a funciton f. The function integral has the following inputs:

f = a function from a float number to a float number

x1 , x2 = the two integration intervals (x1 < x2), float numbers as well

n = the number of intervals (positive integer >0)

The integral is an approximation of the real integral, and it is the sum of the yellow rectangles as shown in the example below, where we use the function f = ½ x, x1=1, x2=20 and n=9. Note that the precise value of the integral is 99.75.



Example of function usage (referred to the above picture):

`integral f 1 20 9 = 89.72222` (the sum of the area of the 9 rectangles with same width).

`integral f 1 20 10000 = 99.75184` (the precision is higher when the number of intervals is higher - and the interval width is smaller. When the number of intervals goes to infinity, the integral value is the correct one).

Note that f (or any other function you want to use) must be defined. In our example f is defined as follows (but it can be any function returning a real number): f x = 0.5*x