MS CONNECT

Assignment 3

Objectives

- Get comfortable with .NET.
- Start thinking more carefully.
- Solve some problems in C#

Reasonable

- Communicating with colleagues about problem problems in English (or some other spoken language).
- Discussing the assignment material with others in order to understand it better.
- Helping a colleagues identify a bug in his or her code, as by viewing, compiling, or running his or her code, even on your own computer.
- Incorporating snippets of code that you find online or elsewhere into your own code, provided that those snippets are not themselves solutions to assigned problems and that you cite the snippets' origins.
- Sending or showing code that you've written to someone, possibly a colleagues, so that he or she might help you identify and fix a bug.

Rules

- Draw the flow chart for each solutions.
- Write the Algorithm for each solution.
- Document your solutions in word file.
- Write proper comments for each line in your source code.
- Document the output of your program.
- Your program should address the problem, there should NOT be any deviations in output.

Question 1

The ratio of the circumference of a circle to its diameter is given by the constant known by the Greek letter pi, and is an irrational number (its representation is non-terminating and non-repeating) with a value slightly larger than 3.14159.

What is the one-thousandth digit of pi? (Counting begins at zero, so the zeroth digit of pi is 3 and the fourth digit of pi is 5.)

Question 2

Students who are learning arithmetic sometimes play a number-guessing game: "I'm thinking of a number between 1 and 100. Can you guess it?" "Is the number less than 50?" "Yes." "Is the number less than 25?" "No." And so on, halving the interval at each step until only one number is left. This technique is known colloquially as the binary chop.

Your first task is to write a function that takes a target number and an array of numbers in non-decreasing order and returns either the position of the number in the array, or -1 to indicate the target number is not in the array. For instance, bsearch(32, [13 19 24 29 32 37 43]) should return 4, since 32 is the fourth element of the array (counting from zero).

Beware that this exercise is harder than it looks. Thus, your second task is to write a suitable test program that shows the accuracy of your binary search function.

Question 2

Find the smallest number that can be expressed as the sum of 7 consecutive prime numbers, the sum of 17 consecutive prime numbers, the sum of 41 consecutive prime numbers, the sum of 541 consecutive prime numbers, and is itself a prime number.

For example, 41 is the smallest prime number that can be expressed as the sum of 3 consecutive primes (11 + 13 + 17 = 41) and the sum of 6 consecutive primes (2 + 3 + 5 + 7 + 11 + 13 = 41).

Your task is to find the number.

Question 3

Words that are formed from the same set of letters are anagrams of each other. For instance, pots, post, stop, spot, opts, and tops are anagrams.

Your task is to write a program that, given a array and an input word, prints all the anagrams of the input word. You are also to determine the largest anagram class in your dictionary.

Question 4

Pig Latin is an English-language word game, usually played by children, in which words are mutated systematically. A word that begins with a vowel (a, e, i, o, or u) has the syllable “way” added to the end; for instance, art becomes art-way and eagle becomes eagle-way. A word that begins with one or more consonants has the initial consonants stripped, moved to the end of the word, then the syllable “ay” is added; for instance, start becomes art-stay and door becomes oor-day. A hyphen is added as shown above as an aid to prevent ambiguity; otherwise, a word like aspray could be the translation of spray (ay-spray) or prays (ays-pray). Even so, some words remain ambiguous; art-way could be translated as either art or wart.

Your task is to write functions that translate English to Pig Latin and Pig Latin to English.

Question 5

My daughter, a freshman in high school, is just completing her first programming class, using C#. Her final assignment was to write a program to print a loan amortization table, given an initial balance, annual interest rate, term in months, and monthly payment.

Your task is to write that program (you are not restricted to Java), then print the amortization table for a three-year car loan of $10,000 at 7%.

Question 6

I was helping my friend with her math homework recently, and came across this problem in continued fractions:

$$1 + \cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{\ddots}}}}}$$

This fraction can be considered as a sequence of terms

$$G_0 = 1$$

$$G_1 = 1 + \tfrac{1}{1} = 2$$

$$G_2 = 1 + \cfrac{1}{1+\cfrac{1}{1}} = \tfrac{3}{2}$$

$$G_3 = 1 + \cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1}}} = \tfrac{5}{3}$$

Or, in general

$$G_{n+1} = 1 + \tfrac{1}{G_n}$$

The first ten elements of the sequence are 1, 2, 3/2, 5/3, 8/5, 13/8, 21/13, 34/21, 55/34, and 89/55.

Your task is to write a program that evaluates the n^{th} element of the sequence. What is the value of G_{200}?

Question 7

In their book *The Practice of Programming*, Brian Kernighan and Rob Pike present this code for matching simple regular expressions consisting of literal characters, a dot matching any character, a star consisting of zero or more repetitions of the preceding character, and a caret and a dollar sign representing the beginning or end of the search string:

```
/* match: search for re anywhere in text */
int match(char *re, char *text)
{
  if (re[0] == '^')
    return matchhere(re+1, text);
  do { /* must look at empty string */
```

```
    if (matchhere(re, text))
      return 1;
  } while (*text++ != '\0');
  return 0;
}

/* matchhere: search for re at beginning of text */
int matchhere(char *re, char *text)
{
  if (re[0] == '\0')
    return 1;
  if (re[1] == '*')
    return matchstar(re[0], re+2, text);
  if (re[0] == '$' && re[1] == '\0')
    return *text == '\0';
  if (*text!='\0' && (re[0]=='.' || re[0]==*text))
    return matchhere(re+1, text+1);
  return 0;
}

/* matchstar: search for c*re at beginning of text */
int matchstar(int c, char *re, char *text)
{
  do { /* a * matches zero or more instances */
    if (matchhere(re, text))
      return 1;
  } while (*text!='\0' && (*text++==c || c=='.'));
  return 0;
}
```

Your task is to port the code to C#. You should use the features and idioms of your language, while simultaneously preserving the beauty of Rob Pike's regular expression matcher.

Question 8

Before computers, arithmetic was done by hand, and one of the primary tools of the practicing arithmetician was a table of logarithms, which was printed in a large book and used to perform multiplication, division and exponentiation. But how to calculate the logarithms?

Leonhard Euler, the Swiss mathematician, explained the algorithm in his 1748 book *Introductio in analysin infinitorum*. The basic idea is that, given two numbers *a* and *b*, $a < n < b$, the geometric mean of *a* and *b* is equal to the arithmetic mean of their logarithms. Thus, to calculate the logarithm of a given number *n*, find two powers of the logarithmic base that bound *n*, calculate the geometric mean (square root of the product) of the two numbers and the arithmetic mean (half of the sum) of the two numbers, and recur on the the smaller interval between one of the two numbers and the geometric mean that bounds *n*, continuing until the desired accuracy is reached.

But how did Euler calculate the square roots? Sir Isaac Newton, the English physicist, described a method based on derivatives. Given an initial estimate *x* of the square root of *n*, a better estimate is given by $x - \frac{x^2 - n}{2x}$. Again, this calculation is iterated until the desired accuracy is reached.

Your task is to write a function to compute square roots using Newton's method, then use that function to compute logarithms using Euler's method.

Question 9

Given an input string and a dictionary of words, segment the input string into a space-separated sequence of dictionary words if possible. For example, if the input string is "applepie" and dictionary contains a standard set of English words, then we would return the string "apple pie" as output.

Here is a number of constraints: The dictionary provides a single operation, exact string lookup, and is a given to the task; you are not to consider how to implement the dictionary, nor or you to worry about stemming, spelling correction, or other aspects of the dictionary. The output may have more than two words, if there is more than one solution you only need to return one of them, and your function should indicate if there are no solutions.

Your task is to write a function that solves the "word break" problem.

Question 10

You are given an array with integers between 1 and 1,000,000. One integer is in the array twice. How can you determine which one?

Your task is to write code to solve the array duplicates problem.

Question 11

In an election, the winner is required to have the majority of the votes. For instance, with the set of votes {A A A C C B B C C C B C C}, the winner is C with 7 of 13 votes. Some elections have no winner; with the set of votes {A B C A B C A}, A gets a plurality of the votes but not a majority, so there is no winner. You can think of voting as a political election, or as redundant hardware in a critical system where a failure of one component must not lead to failure of the overall system.

Your task is to write a function that determines the winner of a vote, or indicates that no candidate reached a majority.

Question 12

Find all possible solutions to the equation 1/a + 1/b + 1/c + 1/d + 1/e = 1 where all of a, b, c, d and e are positive integers.

One solution is the trivial 1/5 + 1/5 + 1/5 + 1/5 + 1/5. Another solution, based on the perfect number 1 + 2 + 4 + 7 + 14 = 28, is 1/2 + 1/4 + 1/7 + 1/14 + 1/28. The minimum distinct solution is 1/3 + 1/4 + 1/5 + 1/6 + 1/20, where all the denominators are distinct and the sum of the denominators 3 + 4 + 5 + 6 + 20 = 38 is minimum over all solutions.

Your task is to write a program to enumerate all possible solutions to the equation.

Question 13

Show how to implement a stack using two queues. Analyze the running time of the stack operations.

You may assume that the queue operations enqueue, dequeue, and isEmpty are provided. You should provide the stack operations push, pop and isEmpty.

Your task is to implement stacks using two queues, as directed above.

Question 14

When I was a kid, telephones had rotary dials, not push buttons, and exchanges had names; my grandmother was in the Underhill 8 exchange. If you were calling someone in the same exchange as you were, you only had to dial the last four digits of the number. Long distance calling generally involved a human operator.

Modern USA telephone numbers have ten digits, segmented as a three-digit area code, a three-digit exchange code, and a four-digit number. Within an area code, you need only dial (the verb hasn't changed, even though telephones no longer have a dial) the seven-digit exchange code and number; otherwise, you must dial the complete ten-digit number, often with a prefix.

Our exercise today asks you to validate a telephone number, as if written on an input form. Telephone numbers can be written as ten digits, or with dashes, spaces, or dots between the three segments, or with the area code parenthesized; both the area code and any white space between segments are optional. Thus, all of the following are valid telephone numbers: 1234567890, 123-456-7890, 123.456.7890, (123)456-7890, (123) 456-7890 (note the white space following the area code), and 456-7890. The following are not valid telephone numbers: 123-45-6789, 123:4567890, and 123/456-7890.

Your task is to write a phone number validator that follows the rules given above; your function should either return a valid telephone number or an indication that the input is invalid.

Question 15

Write a function that takes a string and determines if the delimiters in the string are balanced. The pairs of delimiters are (), [], {}, and <>, and delimiters may be nested. In addition, determine that string delimiters ‘ and ” are properly matched; other delimiters lose their magical delimiter-ness property within quoted strings. Any delimiter is escaped if it follows a backslash.

Your task is to write the function to determine if a string has balanced delimiters.