VICTORIAN CURRICULUM AND ASSESSMENT AUTHORITY

VCE Algorithmics (HESS)

Written examination – End of year

Sample questions

SECTION A – Multiple-choice questions

Question 1

Consider the following pseudocode, written by Cecily, an Algorithmics student.

```
Algorithm MakeChange
      Input: T, integer, total amount of change to be given
      Output: 1-D array, a set of coins to make a total of T

      Let L = an array of coin values in reverse sorted order
      Let finalList = an empty array

      for eachValue in L
            count = ( T / eachValue ) rounded down
            Append eachValue to the end of finalList
            T = T - ( eachValue*T )
      endfor
      return finalList
```

Cecily is checking the pseudocode to solve the change making problem. She uses the following values:

- L = [50, 20, 10, 5, 1]
- T = 175

Which one of the following is an array returned by the pseudocode and is it a correct solution?

A. [50, 20, 10, 5, 1]; the solution is correct

B. [50, 50, 50, 20, 5]; the solution is correct

C. [50, 20, 10, 5, 1]; the solution is incorrect

D. [50, 50, 50, 20, 5]; the solution is incorrect

© VCAA 2017 – April 2017

Question 2

Hilbert's decision problem asks for an algorithm that takes as input a statement of first-order predicate logic and determines whether that statement

A. is universally valid.

B. is algorithmically complete.

C. results in the algorithm returning 'yes'.

D. considers only sets and subsets of natural numbers.

Question 3

The relationship that Big-O, Big-Ω and Big-Θ notations have with each other is

A. $\Theta(f(n))$ necessarily implies $\Omega(f(n))$ but not $O(f(n))$.

B. $O(f(n))$ necessarily implies $\Omega(f(n))$ but not $\Theta(f(n))$.

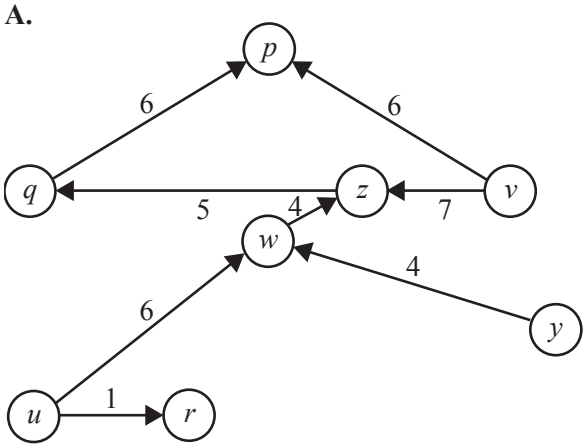
C. $O(f(n))$ necessarily implies $\Theta(f(n))$ and also $\Omega(f(n))$.

D. $\Theta(f(n))$ necessarily implies $\Omega(f(n))$ and also $O(f(n))$.

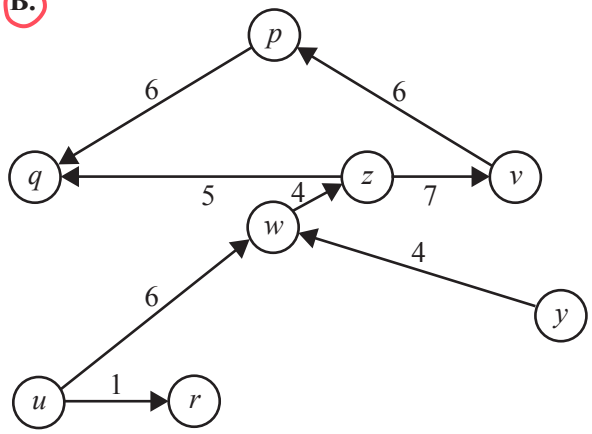
Question 4

Which one of the graphs below is a directed acyclic graph with a topological distance between u and v that is equal to 17?

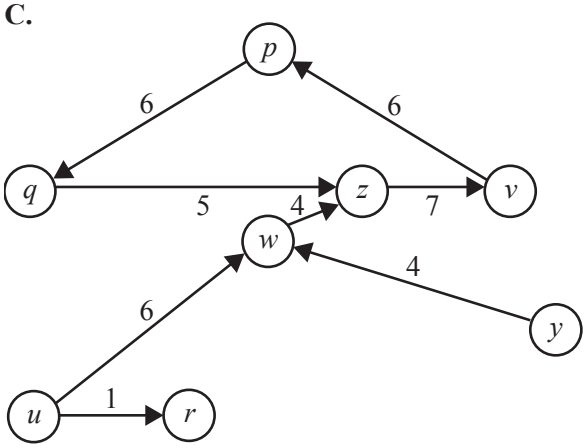
A.



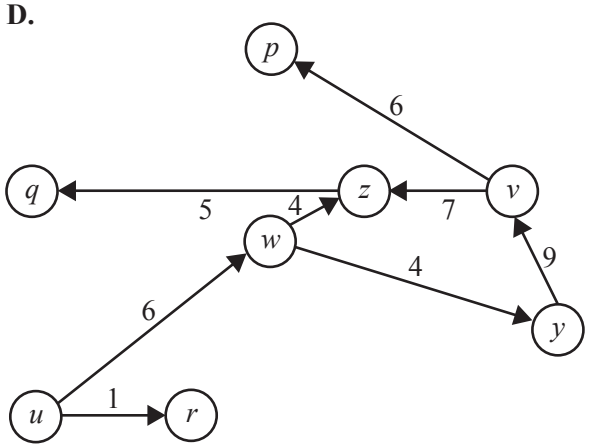
B.



C.



D.



Question 5

A proof by contradiction works on the basis of what logical conclusion?

A. If an assertion is false, its contradiction must be true.

B. If an assertion is contradictory, its opposite must be true.

C. If an assertion necessarily leads to a contradiction, the assertion itself is false.

D. If an assertion necessarily contradicts itself, then the statement has been proven.

Question 6

Consider the BinarySearch algorithm executed on an array consisting of the first 10 numbers of the Fibonacci sequence, where $F_0 = 0$, $F_1 = 1$, $F_2 = 1$, and so on.

```
Algorithm searchFib
      Input: An array A = [F0, F1, F2, …, F9], a target x
      Output: n where x=Fn if target x is in A, -1 otherwise

      low = 0
      high = 9
      while low ≤ high
            mid = ( low + high )/2 rounded down
            if A[mid] > x
                  high = mid - 1
            else if A[mid] < x
                  low = mid + 1
            else
                  return mid
            endif
      endwhile
      return -1
```

0, 1, 1, 2, 3, 5, 8, 13, 21, 34

For which values of x would the algorithm above finish in a best case running time and a worst case running time, respectively?

A. $x = 3, x = 100$

B. $x = 5, x = 100$

C. $x = 5, x = 34$

D. $x = 0, x = 0$

© VCAA 2017 – April 2017

SECTION B

Question 1 (4 marks)

Bernie is a professional computer gamer who competes against other players around the world in gaming competitions. Every time he plays against a competitor, he documents information about how they play and how well they do, and stores this information in a file, using the competitor's unique gamer name as the filename. Each file contains information on a single player, such as play styles, win and loss rates, and optimal conditions for successful play.

Bernie has been playing in competitions for a very long time and he now has more than 50 000 separate files documenting the players he has competed against in the past.

Bernie's computer does not have in-built functions for searching folders or files. He would like to be able to search a folder for a file matching a gamer's name so he can quickly look up a player and view or add to the information he has collected on this player. He plans to use binary search to do this, but must first sort the files in alphabetical order for binary search to work.

Explain how a sorting solution using a divide and conquer algorithm design pattern would allow Bernie to efficiently sort the players' filenames.

- Bernie can use an efficient divide & conquer sorting algorithm MergeSort
- The algorithm would split the list of players' names in half recursively until there is one element At this point, the algorithm will merge the lists, sorting and combining them in linear time For each of the log n splits, the algorithm merges the lists in linear time, giving $T(n) \in O(n \log n)$

Question 2 (3 marks)

Distinguish between time complexity and space complexity. Use an example as part of your response.

-Time complexity is a measure of an algorithm's running time and operations executed in relation to its input size, while space complexity measures memory storage An example is storing all integers 1 to n in a list takes $O(n)$ time to perform, and because the list contains n elements, the space complexity is $O(n)$

Question 3 (4 marks)

Cobham's thesis asserts that only tractable computational problems can be computed on a computational device.

Discuss the limitations of this assertion in relation to practical feasibility.

- Cobham's Thesis defines tractability as solvable in polynomial time However, for most computers, an algorithm running in $O(n^{1000})$ for example would still be infeasible to compute for large inputs, while an algorithm in $O(2^n)$ could prove more practical
- However, Cobhams Thesis would deem an exponential time algorithm intractable despite it requiring an enormous input to outgrow $O(n^{1000})$ It is limited because it only considers the feasability of an algorithm as the input size approaches infinity

Question 4 (7 marks)

A word is described as 'alphabetically distant' if, when all its letters are rearranged in alphabetical order, no two neighbouring letters are also neighbouring in the alphabet. Repeated letters are also considered neighbouring. Some examples are provided below:

- 'shape' is alphabetically distant as 'aehps' does not have any two neighbouring letters that are also neighbouring in the alphabet.
- 'algorithm' is not alphabetically distant as 'aghilmort' has three pairs of letters that are neighbouring in the alphabet ('gh', 'hi', 'lm').
- 'program' is not alphabetically distant as 'agmoprr' has a repeated letter ('rr').
- 'maze' is alphabetically distant as 'aemz' does not have any two neighbouring letters. (Note that 'az' is not considered neighbouring.)

Jasmine has been tasked with designing an algorithm that scans a given text to find the longest alphabetically distant word. To make the task easier, she has been given all of the words in the text as arrays containing their letters' positions in the alphabet, sorted in ascending order. For example, 'shape' is given as [1, 5, 8, 16, 19], where the first element has an index of 0. Jasmine has designed the following algorithm.

```
Algorithm AlphaDistant
    Input: text, list containing n arrays each representing a single
    word as integers, sorted ascending
    Output: integer, the length of the longest alphabetically distant
    word

    Let longestword = 0
    for word in text
        for i = 0 to length of word - 1
            if word[i] = word[i+1] or word[i] - word[i+1] = 1
                return FALSE
            else
                if length of word > longestword
                    Let longestword = length of word
                endif
            endif
        endfor
    endfor
    return longestword
```

a. Annotate Jasmine's algorithm above to describe **three** errors present in her attempt. 3 marks

b. Rewrite Jasmine's algorithm so that it will correctly solve the errors. 4 marks

```
Let longestWord = 0
for word in text
    for i = 0 to length of word - 2
        if word[i] = word[i+1] or word[i+1] - word[i] = 1
```

break

if length of word > longestWord

longestWord = length of word

return longestWord

Question 5 (2 marks)

Explain why, when demonstrating the correctness of an algorithm by induction, it is not sufficient to simply demonstrate that the inductive step holds.

- An inductive proof requires a base case to be shown true as well, as without establishing one would not show what set of inputs the algorithm is correct for. This is why a base case is essential

Question 6 (8 marks)

An *n* metre long roll of silk fabric is to be cut and sold. The value of each sheet of fabric is dependent on its length. The table below provides the values for the three smallest possible lengths, but all values for each length between 1 and *n* are known.

Length of sheet of fabric	Value
1 m	$2.10
2 m	$3.40
3 m	$6.55

a. Outline a greedy heuristic approach to determine how the *n* metres of fabric should be cut in order to maximise the total value of the fabric. 4 marks

- A greedy heuristic approach for this problem would be to take the $/m for each fabric, i.e, $1.70/m for the 2m length, and repeatedly cut by the maximum $/m rate for k length. When the remaining length < k, take the greatest $/m rate for less than k length and repeat until the fabric has been optimally cut by the greedy approach

b. Outline a randomised heuristic approach to determine how the *n* metres of fabric should be cut in order to maximise the total value of the fabric. 4 marks

Hill-Climbing generate a random solution to start Make a slight, random modification to the way in which the fabric is cut - i e, changing a 5m cut to 2m & 3m or vice versa, and then check if the value obtained from the modified solution is > than the previous' If so, accept it as the new solution and repeat for a certain number of iterations Final solution will be approximate

Answers to multiple-choice questions

Question	Answer
1	C
2	A
3	D
4	B
5	C
6	A

© VCAA 2017 – April 2017