2016 VCE Algorithmics (HESS) examination report

General comments

The 2016 Algorithmics (HESS) exam contained two sections. Section A was composed of 20 multiple-choice questions and Section B was composed of 17 short-answer and extended-answer questions. Students achieved scores across the range of available marks, with higher-scoring students demonstrating an impressive grasp of challenging content.

Time management was a key concern for some students. Some students provided responses that were longer than required for questions in Section B – particularly Questions 1, 2 and 3 – and questions later in the examination were not attempted or appeared rushed. Students are encouraged to use the reading time well and to consider the marks allocated for each question to ensure that they are not spending a disproportionate amount of time on one question. Students do not need to complete the examination in sequential order.

A number of questions on the examination asked students to 'explain how' in reference to a concept (Questions 1, 3 and 6d.) or similarly, to 'explain why' (Questions 7b. and 9). Students needed to carefully read the questions and respond appropriately – only reiterating the property and not explaining it was not addressing the question. For instance, in answering Question 3, many students simply wrote that the divide-and-conquer approach would be faster than the linear search approach. It is important that students are aware of the specific implications of the various command terms used throughout an examination and structure their answer accordingly.

Another common source of error was a tendency to answer questions generically rather than addressing the specific context given in the question. For example, a number of students were able to name two valid approaches in response to Question 8, but then proceeded to simply define those approaches generally rather than describe how they would be used with respect to the context of the block structure.

Some students were able to write algorithms in pseudocode in response to unseen contexts (Questions 15a., 15b., 15c., 16a. and 16c). Students should, however, note that adhering too closely to the syntax of a programming language is discouraged as it can result in unnecessary complications in how the algorithm is structured and hinder the clarity of their description. Many responses could have been improved by greater use of natural language.

Specific information

Note: Student responses reproduced in this report have not been corrected for grammar, spelling or factual information.
This report provides sample answers or an indication of what answers may have included. Unless otherwise stated, these are not intended to be exemplary or complete responses.

The statistics in this report may be subject to rounding resulting in a total more or less than 100 per cent.

Section A – Multiple-choice questions

The table below indicates the percentage of students who chose each option. The correct answer is indicated by shading.

Question	% A	% B	% C	% D	% No Answer	Comments
1	12	4	44	38	1	Students should note that the while loop will still be entered if *i* is equal to *n*, and so they must execute the two lines inside the loop before stopping.
2	4	84	7	4	0	
3	74	19	1	6	0	
4	3	60	12	25	0	
5	13	6	10	71	0	
6	0	78	7	13	1	
7	4	4	81	7	3	
8	0	4	90	6	0	
9	51	13	10	25	0	It is crucial that students understand the implication of *distinct* edge weights in minimum spanning tree questions. Namely, it is true that a graph with *distinct* edge weights will necessarily have a unique spanning tree. The edge e_{max} will be part of this minimal spanning tree if there is no other path between its two incident vertices, and therefore option A was false.
10	21	16	59	4	0	Negative weight edges remove the guarantee that Dijkstra's will report correct shortest distances to destination nodes. Their presence, however, does not guarantee that Dijkstra's will not report correct shortest distances.
11	16	7	69	7	0	
12	79	16	1	3	0	
13	31	21	15	34	0	Students may find it helpful to consider the proof of Bellman-Ford's correctness in understanding its function.

Question	% A	% B	% C	% D	% No Answer	Comments
14	28	1	16	54	0	It is important for students to note that the number of outbound links from a webpage does not affect its own PageRank whatsoever; it merely affects its contribution to the PageRank of adjacent nodes.
15	6	31	3	60	0	
16	13	3	74	10	0	
17	69	13	1	16	0	
18	6	75	16	3	0	
19	21	62	10	7	0	
20	63	3	10	24	0	Students must not confuse the dequeue operation in a high-level programming language like Python, which will return an element, with the dequeue operation as defined by the signature specification, which will return a queue.

Section B

Question 1

Marks	0	1	2	3	Average
%	20	14	37	29	**1.8**

Most responses identified that a heuristic can help find a suboptimal or ‘good enough’ solution quickly, thereby overcoming the soft limits of computation. For full marks, responses needed to discuss the advantages of randomisation generally or clearly describe how a specific randomised heuristic functions to overcome soft limits.

The following is an example of a high-scoring response.

> *Randomised heuristics can be used to give an approximate solution to a problem much faster than an exact solution can be found for some problems. E.g. in the travelling salesman problem, generating a random path then improving it with simulated annealing will yield an approximate and usable solution.*

Question 2

Marks	0	1	2	3	Average
%	24	27	34	15	**1.4**

Two interpretations of the question were accepted. Responses could have described an approach for verifying a proposed structure or described an approach for discovering a valid structure. Graph colouring is one approach that could be used to ensure that no two adjacent components in a proposed structure are of the same type.

Where the problem was understood as being equivalent to an NP-Complete problem (as in the case of graph colouring), responses needed to describe a heuristic (e.g. greedy colouring) that would solve the problem in a reasonable time.

Question 3

Marks	0	1	2	3	Average
%	10	22	42	26	**1.9**

Responses needed to describe a divide-and-conquer (or decrease-and-conquer) approach that would progressively halve the search space until the lightest diamond in a particular batch is found. This would solve the problem in $O(\log n)$ time rather than the $O(n)$ time that the brute force approach requires.

While a number of responses correctly noted that this approach was analogous to a binary search algorithm, responses needed to distinguish between algorithm design patterns, such as divide-and-conquer, and algorithms, such as binary search.

Students are also encouraged to pay careful attention to the word 'how' with respect to the phrasing of questions. Most responses correctly stated that the divide-and-conquer approach would solve the problem more efficiently, but this was insufficient unless they also clearly articulated either the Big-O running time of the improved approach or the idea of halving the search space each time.

Question 4

Marks	0	1	2	Average
%	8	10	82	**1.8**

A priority queue would allow for the urgent packets, such as voice and video, to jump to the front of the queue for processing and forwarding before less urgent traffic.

This question was answered well, but students must take care to answer questions of this type with respect to the specific context.

Question 5

Marks	0	1	2	Average
%	23	16	61	**1.4**

During a depth-first traversal, a list of visited nodes could be kept. When the traversal terminates, if there are any unvisited nodes in the graph, then these are disconnected.

Some responses provided a good method, but did not adequately articulate how their method would conclude whether the graph *G* was disconnected.

Question 6a.

Marks	0	1	2	Average
%	34	40	26	**0.9**

The following is an example of a high-scoring response.

> *d is the damping factor, and represents the probability of a web surfer following links on a webpage, instead of jumping to a random webpage.*

Question 6b.

Marks	0	1	Average
%	68	32	**0.3**

(1 – *d*) represents the probability of the surfer clicking on a random page rather than following a link. As there are N websites, (1 – *d*)/*N* represents the probability of a surfer randomly clicking on a given website.

Question 6c.

Marks	0	1	Average
%	62	38	**0.4**

The probability of an imaginary surfer following links on existing pages to arrive at the page

Question 6d.

Marks	0	1	2	Average
%	74	22	4	**0.3**

As node *E* has no outbound links, its PageRank contribution should be distributed evenly among all other nodes. This can be achieved by adding an imaginary link from *E* to every other node in the graph for the purposes of calculation.

While many responses correctly identified that the absence of outbound links creates a ‘sink’, which is problematic for PageRank calculations, this answer alone did not deal with the question of **how** PageRank would include *E*.

Students should be discouraged from answering questions that ask for a method with a negative (that is, claiming *E* would simply not be included).

Question 7a.

Marks	0	1	Average
%	24	76	**0.8**

Any nonempty subset of *C*, *F*, *J*, *I*, *G*, *H* was accepted as the exact answer varied based on the order in which edges are checked.

Question 7b.

Marks	0	1	2	Average
%	53	17	31	**0.8**

The Bellman-Ford algorithm will report the correct shortest distance from the source to a node, contingent on the absence of a negative weight cycle in any path between the source and that node. Nodes *C*, *F*, *J*, *I*, *G* and *H* were all reachable by a path containing a negative weight cycle, whereas nodes *B*, *D* and *E* were not. Thus, only the latter will necessarily remain the same distance from source *A*, as the distances to those nodes are correct after nine iterations.

Question 8

Marks	0	1	2	3	4	Average
%	15	11	32	17	26	2.2

Some responses conceived of one approach but not two. Exploring different ways to solve a particular problem throughout the year would be a good way to develop capacity for this type of question. Responses should have outlined approaches in a context-specific way, rather than discussing approaches generically.

The following is an example of a high-scoring response.

Backtracking. Start with one block. Repeatedly place new blocks on connectors. If a sculpture is invalid, backtrack by removing the most recently added blocks. Stop when every cell of the 8x8 square is covered.

Divide and conquer. The problem can be simplified to a 4x4 version, and simply repeated 4 times, covering the 8x8 square.

Question 9

Marks	0	1	2	3	4	Average
%	43	25	12	8	12	1.2

While there was a general familiarity with the travelling salesman problem, responses often lacked precision.

Students should ensure that they are able to differentiate between the decision and optimisation versions of a given problem. There was also some confusion as to what constituted an NP problem, with many students discussing the time taken to solve the problem, rather than the time taken to verify a solution.

The following is an example of a high-scoring response.

The decision version of the travelling salesman problem says:

Given a complete graph G, does there exist a path in G, which starts at a certain node, and finishes at the node, visiting all other nodes in G exactly once, that has a total weight smaller than a number X?

The problem is NP as, given a solution to the problem, it is able to be verified in polynomial time. Simply sum all the weights of edges in the path and check if it is indeed smaller than X.

Question 10

Marks	0	1	2	3	4	Average
%	37	9	9	5	39	2

$T(n) = 3T\left(\frac{n}{3}\right) + O(n)$

$a = 3, b = 3$ and $d = 1$

So $a = b^d$ (Case 2)

$\therefore T(n) = O(n \log n)$

Of those students who demonstrated an understanding of the Master Theorem, the most common mistake was simply assuming that mergesort always uses two sublists rather than the three stated in the question.

Question 11a.

Marks	0	1	2	Average
%	66	31	3	**0.4**

0.01 < cooling_factor < 1

A number of responses correctly identified one boundary but ignored the other. A less common source of error was the inclusion of the endpoints within the allowed range of values.

Question 11b.

Marks	0	1	Average
%	54	46	**0.5**

Sometimes replacing soln with soln_new even when it is not an improvement allows for the possibility of escaping local optima to reach a global optimum.

Question 11c.

Marks	0	1	2	Average
%	61	15	25	**0.7**

There were many acceptable answers, including sudoku, scheduling, travelling salesman problem, etc. Any NP-Hard example that could not be easily solved by brute force was accepted.

Some responses identified a suitable problem but did not discuss how cost(soln) was a reasonable one. Cost functions that were flat for most of their domain were not suitable.

Question 12

Marks	0	1	2	3	Average
%	16	37	27	20	**1.5**

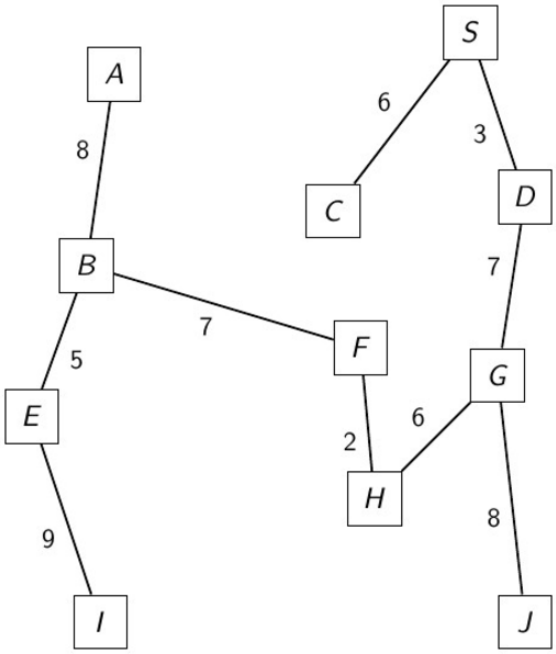
A tree that clearly captured the idea of turn-taking and abided by the rules of the game was required. Responses did not necessarily have to involve drawing the entire game tree; drawing the entire subtree with root (1,1,1) sufficed to determine the move Player A should make.

It was evident that some students misread the question and instead addressed the question of which move Player A should make.

Other responses included poorly presented game trees, leading to mistakes. Responses for this type of question should be completed in pencil to allow for amendments and refinements where needed.

Question 13a.

Marks	0	1	2	Average
%	15	42	44	**1.3**



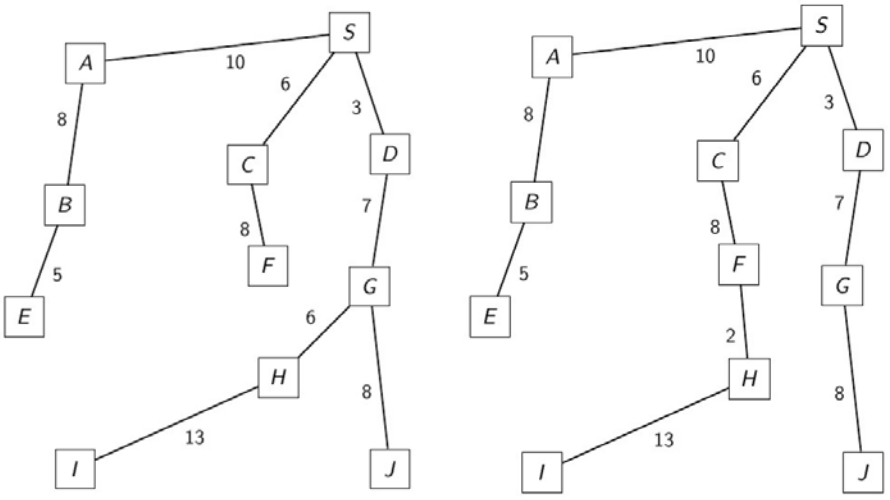
Conditions met: Condition 1

The minimum cost spanning tree produced by Prim's algorithm was generally drawn well, although some students made minor errors or gave the incorrect subsequent condition.

Question 13b.

Marks	0	1	2	Average
%	50	37	12	**0.6**

There were two possible responses:



Option 1

Option 2

For Option 1, only Condition 1 was met.

For Option 2, both Conditions 1 and 2 were met.

There was some confusion regarding how Dijkstra's algorithm would produce a tree, with some students adding an edge from the source node to every other node. The tree produced should have shown the shortest path from the source to every other node.

Question 13c.

Marks	0	1	2	3	Average
%	61	18	21	1	**0.6**

Condition 1 is met by Dijkstra's algorithm without any modifications. For Condition 2 to be met, the shortest path tree generated by the modified algorithm must have a total weight less than or equal to the total weight of all possible shortest path trees.

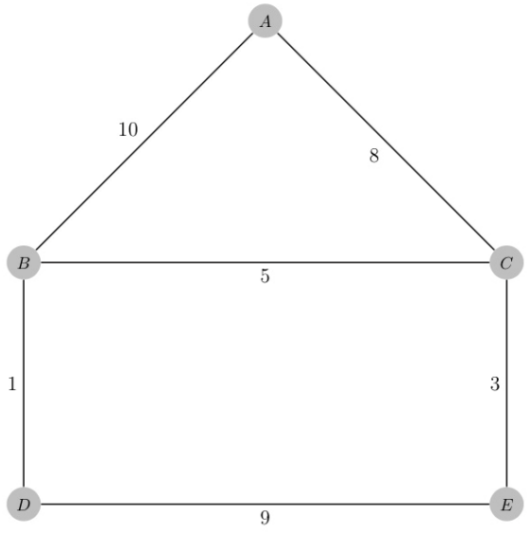
Dijkstra's standard algorithm randomly selects the next nearest node to the source where there are two or more possible choices with equal distance from the source. For Condition 2 to be met, where there are two or more options for the next nearest node from the source, the node with the least weight edge leading to it should be selected.

Students are encouraged to practise describing the process of an algorithm in words, as many responses were undermined by an inability to describe Dijsktra's algorithm, thereby rendering the modification nonsensical.

Additionally, it was evident that a significant number of students had read Condition 2 to mean that the graph needed to be both a shortest path tree and a minimum spanning tree. Based on this interpretation, there were some responses that were able to explain why no modification could exist that would allow for Condition 2 to be met. However, many students dismissed the possibility simply because Dijkstra's algorithm does not return a minimum spanning tree, or similar. This does not address the question of whether a modification exists.

Question 14a.

Marks	0	1	2	3	Average
%	2	4	25	69	**2.6**



The schools are represented as nodes.

The distances are represented as edge weights.

Question 14b.

Marks	0	1	Average
%	18	82	**0.8**

20 km

Question 14c.

Marks	0	1	Average
%	10	90	**0.9**

A–B–D

Question 14d.

Marks	0	1	Average
%	13	87	**0.9**

17 km (the total weight of the minimum weight spanning tree)

Question 14e.

Marks	0	1	2	Average
%	36	20	45	**1.1**

Some responses described how a particular edge could be amended but did not extend their idea to the general case.

The following is an example of a high-scoring response.

Dijkstra's may be used by modifying the weight of each edge such that weight = 2 x weight + 1.

Question 15a.

Marks	0	1	2	Average
%	21	55	24	**1**

The following is an example of a high-scoring response.

```
washClothes(basket):
     if basket size = 1:
          wash single item of clothing
     else:
          washClothes(first smaller basket)
          washClothes(second smaller basket)
```

Most responses used a recursive approach with some division of clothes, but many forgot to include the base case. Responses require a base case for any recursive approach if the algorithm is to terminate.

Question 15b.

Marks	0	1	2	3	Average
%	20	14	31	35	1.8

Both iterative and recursive approaches were well represented.

The following is an example of a high-scoring response.

```
findMatch(sock, sockList)
      for each candidate_sock in sockList:
            if candidate_sock matches sock:
                  return (sock, candidate_sock)
      return nothing
```

Question 15c.

Marks	0	1	2	3	Average
%	54	16	15	15	0.9

```
Let pairs be an empty list
match(listOfSocks, pairs)
      if listOfSocks is empty
            return pairs
      Let a be the first sock in listOfSocks
      Pair(a,b) = findMatch(a, listOfSocks)
      add Pair(a,b) to pairs
      listOfSocks = removePair(Pair(a,b), listOfSocks)
      return match(listOfSocks, pairs)
```

Some students did not use any form of tail-recursive approach, while others attempted to define one or both of `findMatch` and `removePair`, usually leading to errors.

Question 16a.

Marks	0	1	2	3	4	Average
%	13	10	28	31	18	2.3

The majority of students were able to access some marks in this question, showing good conceptual understanding of what the question required.

The most common errors involved students not using correct operations for the given data type (stack). Students needed to use a combination of *push*, *top* (or *peek*) and *pop* in the algorithm, rather than generic terms such as ‘move’, ‘shift’ or similar. Some students who attempted to use stack operations did so incorrectly; for instance, pushing straight from one stack to another without first popping.

Another common error was moving papers between stacks without first *doing* and *marking* them.

The following is an example of a high-scoring response.

```
While incomplete_stack is not empty:
      pop incomplete_stack and do test paper
      push paper onto complete_stack
While complete_stack is not empty:
      pop complete_stack and mark test paper
      push paper onto marked_stack
```

Question 16b.

Marks	0	1	Average
%	18	82	**0.8**

12

Question 16c.

Marks	0	1	Average
%	20	80	**0.8**

2*n*

Question 16d.

Marks	0	1	2	3	Average
%	23	26	39	12	**1.4**

Some students were able to find the highest score in the pile, but to receive full marks responses needed to identify the highest-scoring test itself.

The following is an example of a high-scoring response.

```
pop marked_stack onto desk
repeat (number_of_papers – 1) times
      if top of marked_stack better result than desk paper
            push desk paper to complete_stack
            pop marked_stack to desk
      else
            push desk paper to incomplete_stack
            pop marked_stack to desk
            push desk paper to complete_stack
            pop incomplete_stack to desk
At the end the best paper will remain on Trudi’s desk
```

Question 17

Marks	0	1	2	3	4	5	6	Average
%	15	12	13	15	16	12	18	**3.1**

The Systems Reply was the most commonly cited response and was discussed with varying levels of success. Some students were not able to name a second response or simply named it without demonstrating any level of understanding.

Other standard replies discussed included the Brain Simulator Reply, the Other Minds Reply, the Intuition Reply and the Virtual Mind Reply. Discussing Searle's rebuttals of the replies was not required.

Students are cautioned against conflating the Brain Simulator and Virtual Mind replies with the Systems Reply. Some students compared the man in the room with neurons, and the room itself with the brain; this is a good metaphor to use, but it supports the Systems Reply and not the Brain Simulator Reply. Similarly, the Virtual Mind Reply requires a discussion of the generation of a new, virtual understanding – simply describing the room and the man together as a virtual mind is not describing the Virtual Mind reply.

The following is an example of a high-scoring response.

> *System's Reply response: states that while the person inside the room may not understand Chinese, the system as a whole (room and person and instructions) does understand. Much like individual neurons not understanding but the brain as a whole does.*
>
> *Robot reply: what is preventing the man in the room from understanding Chinese is the sensory motoric disconnect between him and the outside world. If we attach sensors as input methods and robotic arms and legs as output methods, then the man is able to attach meaning to the Chinese characters by interacting with the world. This is exactly how babies learn language.*