**GRIFFITH COLLEGE DUBLIN**

**GRIFFITH COLLEGE CORK**

**GRIFFITH COLLEGE LIMERICK**

**QUALITY AND QUALIFICATIONS IRELAND**

**EXAMINATION**

**HIGHER CERTIFICATE IN COMPUTING**

**STAGE II**

**DATA STRUCTURES AND ALGORITHMS**

**Module Code: HCC-DSA**

**BACHELOR OF SCIENCE IN COMPUTING**

**STAGE II**

**DATA STRUCTURES AND ALGORITHMS**

**Module Code: BSCO-DSA**

**BACHELOR OF SCIENCE (HONS) IN COMPUTING SCIENCE**

**STAGE II**

**DATA STRUCTURES AND ALGORITHMS**

**Module Code: BSCH-DSA**

**Lecturer(s): Tony Mullins**/**Aqeel Kazmi**

**Noel Lynch/Elaine Tynan**

**External Examiner(s): Dr Aidan Mooney**

**Dr Keith Maycock**

##### Date: 12th May 2015 Time: 9.45 – 11.45

# THIS PAPER CONSISTS OF TWELVE QUESTIONS

# TEN QUESTIONS TO BE ATTEMPTED

# ALL QUESTIONS CARRY EQUAL MARKS

**QUESTION 1**

1. Given below is a recursive function that calculates power(a,n). By evaluating (executing) this function show that power(2,4) = 16.

static int power(int a, int n){

if(n == 1)

return a;

else

return( a\* power(n-1));

}

**(5 marks)**

1. Write a recursive function that takes an integer array as one of its arguments and computes the sum of the numbers in the array.

**(5 marks)**

**Total (10 marks)**

**QUESTION 2**

1. Using the statement execution times defined for HAL (See **Appendix** at the end of the exam paper), calculate the running time of the given code fragment.

int f[] = new int[100];

for(int j = 0; j < f.length; j = j + 1) f[j] = j\*9;

int sum = 0;

for(int j = 0; j < f.length; j = j + 1){sum = sum + f[j];}

**(5 marks)**

1. Using the laws of *big O* prove that the computation time of the given code fragment is *O(N)*. (The laws are given in the **Appendix** to this exam paper.)

int f[] = new int[N];

int j = 0;

while(j < N){

f[j] = (int)(Math.random()\*100);

j = j + 1;

}

j = 0;int sum = 0;

while(j < N){

sum = sum + f[j];

j = j + 1;

}

**(5 marks)**

**Total (10 marks)**

**QUESTION 3**

1. To prove that *f(n)* is *O(g(n))* we show that . Using this method prove that , is *O(n)* and that , where a, b are constants, is O(n2).

**(5 marks)**

1. Calculate a cost function for the given code fragment and show that it is O(N2).

int a = 0;

while(a < N){

int b = 0;

while(b < N) b = b + 1;

a = a + 1;

}

**(5 marks)**

**Total (10 marks)**

**QUESTION 4**

1. Write a function that sorts an array of integer values. You may use any sorting algorithm you have studied.

**(7 marks)**

1. Analyse the performance of your chosen sorting function and contrast it with any other sorting function you have studied in your course.

**(3 marks)**

**Total (10 marks)**

**QUESTION 5**

1. *Dynamic data structures are preferred over fixed size data structures and generic data structures are preferred over fixed type data structures when designing a data structure.*

Why do you think this is the case?

**(4 marks)**

1. Explain, with the aid of diagrams, how a hash table can be used to optimise the cost of insertion and retrieval for data collections.

**(6 marks)**

**Total (10 marks)**

**QUESTION 6**

1. Draw a diagram of a sorted doubly linked list of the elements 1,2,3,4,7,9. Insert the number 6 in your original list so that the ordering is preserved.

**(4 marks)**

1. Given below are two classes, Node and LinkedListInt. Class Node implements a node in a linked list and class LinkedListInt implements a singly linked list of nodes where new nodes are inserted at the head of the list. Your task is to write two methods for this class. The methods are: remove the element at the head of the list, if any and size that returns the number of elements in the list.

class Node{

int data;

Node next;

public Node(int x){data = x; next = null;}

public Node next(){return next;}

public void setNext(Node p){next = p;}

public int data(){return data;}

}

class LinkedListInt{

Node head = //head of list

public void remove(){ }

public int size(){ … }

}

**(6 marks)**

**Total (10 marks)**

**QUESTION 7**

1. A queue is a *first in, first out* linear data structure. Queues are typically characterized by three methods: join, leave, and head. Explain the semantics of each of these operations.

**(3 marks)**

1. Given below is an interface for a generic queue. Provide an implementation of this interface using an appropriate class from the Collections library or a dynamic array.

interface Queue<E>{

public boolean join(E x);

public boolean leave();

public E head();

public boolean empty();

}

**(7 marks)**

**Total (10 marks)**

**QUESTION 8**

1. State the definition of a binary search tree and, using a diagram, insert the following list of elements in a binary search tree:  *11,6,13,4,7,10,15,14,5*.

**(4 marks)**

1. Delete the node with the number 6 from your binary search tree in part (a).

**(3 marks)**

1. Show that creating a binary search tree with a sorted list of values results in a linear linked list.

**(3 marks)**

**Total (10 marks)**

**QUESTION 9**

1. Given below is class BST that implements a binary search tree whose elements are integers. The attribute root refers to the root node of the tree (class BNode is also given). This class has two public methods add that could be used to add new values to the tree and sum that calculates the sum of the elements in the tree. Your task is to write the private method sum(root) that actually performs the calculation.

class BST{

private BNode root;

public BST(){root = null;}

public void add(int x){root = add(root,x);}

int sum(){return sum(root);}

}

class BNode{

private int data;

private BNode left;

private BNode right;

public BNode(int d){data = d; left = null; right = null;}

public int data(){return data;}

public BNode left(){ return left;}

public BNode right(){return right;}

public void setLeft(BNode k){left = k;}

public void setRight(BNode k){right = k;}

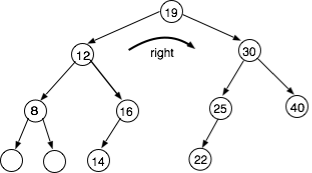
}

**(5 marks)**

1. Given below is a binary search tree. Your task is to perform a right rotation at the root of the tree that preserves the ordering of the tree.

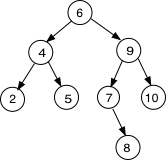
**(5 marks)**

**Total (10 marks)**



**QUESTION 10**

1. State the definition of a balanced binary search tree and show that the given avl tree is balanced.

**(4 marks)**

1. Insert the given list of values in an *avl* tree. The list is: 6, 2, 3, 1, 0, -1.

**(6 marks)**

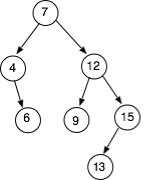
**Total (10 marks)**

**QUESTION 11**

1. Construct a B-tree with a maximum of 4 items per node for the list of numbers: *20, 16, 1, 12, 4, 7, 13, 25, 9, 14, 23, 34, 2, 0*.

**(5 marks)**

1. Explain how a queue can be used to perform a *breadthFirst* traversal of a binary search tree. Use a queue to perform a *breadthFirst* traversal of the given tree.

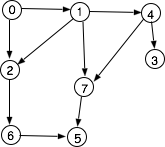


**(5 marks)**

**Total (10 marks)**

**QUESTION 12**

1. Given below is a graph. List the vertices of the graph, the edges in the graph and state the order of the graph. What is the degree of 1 in the graph?



**(5 marks)**

1. Draw a picture of an adjacency list data structure that could be used to represent this graph.

**(5 marks)**

**Total (10 marks)**

**Appendix**

|  |  |
| --- | --- |
| Calculating Running Times on HAL | |
| **Statement** | **Unit cost (ns)** |
| -, \*, /, %, ^, <, >, ==, >=, <=, !=, = | *10ns* |
| Function invocation | *50ns* |
| Argument passing | *10ns* per argument |
| return | *50ns* |
| if(b) s1; else s2 | the cost of b plus the max cost of s1, s2 |
| for, while loops | *totalCost = cost of initialization of variables +*  *(n+1) \* cost of evaluating guard on loop*  *+*  *n \* cost of executing loop body,*  *where n equals the number of iterations of the loop.* |
| New | *100ns* |
| Calculating array indices | *50ns* |
| Math.random() | *100ns* |

**Laws of *big-O***

The laws of *big-O are*:

1. **Summation**

*O(1)+O(1)+..+O(1) = k \* O(1) = O(1)*, where *k* is a constant.

*O(n) + O(n)+..+O(n) = k \* O(n) = O(n)*, where *k* is a constant

*O(n) + O(m) = max(O(n), O(m))*

e.g. *O() + O( ) = O( )*

1. **Product**

*O(n) \* O(n) = O*

*n \* O(n) = O*

*O(n) \* O(m) = O(n \* m)*

*O(k \* f(n)) = k \* O(f(n)) = O(f(n))*, where *k* is a constant

*O() \* O() = O()*

The *big-O* sets of order functions form a chain of sub-sets as follows: