OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

Autumn Examinations 2022-2023

Course Instance Code(s)	1OA2, 3BCT1, 3BS9, 4BS2
Exam(s)	3rd Year Examination Computing Science and IT
Module Code(s)	CT331
Module(s)	Programming Paradigms
Paper No.	1
External Examiner(s)	Dr. Ramona Trestian
Internal Examiner(s)	Professor Michael Madden *Dr. Finlay Smith

Instructions: Answer any 3 questions. All questions will be marked equally. Each question is worth a maximum of 25 marks. The total (out of 75) will be converted to a percentage after marking.

Duration	2 hours
No. of Pages	5
Discipline(s)	Computer Science
Course Co-ordinator(s)	Dr Colm O'Riordan

Release in Exam Venue	Yes [ ]	No [X]
MCQ Answersheet	Yes [ ]	No [X]
Handout	None	
Statistical/ Log Tables	None	
Cambridge Tables	None	
Graph Paper	None	
Log Graph Paper	None	
Other Materials	None	
Graphic material in colour	Yes [ ] No [X]	

PTO

1)

a) What are the two ways that memory management is handled in 'C'? Describe the advantages and disadvantages of both of them.
(8 marks)

b) In your own words describe the effect of the following functions in 'C'.
i) malloc()
ii) calloc()
iii) free()
(3 marks)

c) What is the difference between *void** and other pointer declarations? Give a simple example, including defining and assigning a value to a variable of type *void**.
(3 marks)

d) Write 'C' code that defines a structured type called *lectureRoom* with members that store the name of the room as a character array, the names of the modules that use the room as an array of pointers to strings and an array of pointers to integers that represents the number of students that take each of the modules and an integer which stores the number of modules that use the room. Write a function called *deleteRoom* that accepts a pointer to a *lectureRoom* instance and frees all of the memory associated with the structure.
(11 marks)

2)

a) How do you define and use a function pointer in 'C'? Illustrate your answer with code snippets.
(5 marks)

b) Write a generic function in 'C' that prints the contents of an array of 20 elements. Your function should be passed a pointer to a print function that prints the elements of the type contained in the array (e.g., print an integer etc.) and then prints the elements in the array using that function. You should write a print function named printInt, and finally show you would call your generic function using this print function. What changes would be needed if the array was to contain floating point numbers instead of integers?
(20 marks)

PTO

3)

a) How does Lisp differ from other programming languages (such as 'C'). What are the fundamental operations that are written in Lisp?
(3 marks)

b) In Lisp, what would the following function calls return?

i) (car (cdr (cdr (cdr '(2 3 4 5 6)))))
ii) (car (car '((2 1 1 2))))
iii) (car (car (cdr '(h (7 4) y y z)))
iv) (car (cdr (cdr '((4 (g l) a l) n p 3))))
(4 marks)

c) In Lisp, use car and cdr to return the following – in each case you can assume the lists have been stored in a variable called lst:
i) Element 9 in the list '(s (m p) 3 (3 9))
ii) List (7) in the list '(8 (((7) 5)) p w)
iii) Element a in the list '(2 (9 (a 3) 8) c)
iv) Element '(1 2) in the list '((5 6) (1 (((1 2)))))
(4 marks)

d) Write a non-tail recursive function in Scheme which takes 2 arguments (both lists) and returns a list of all of the numbers from the first list that are also on the 2nd list – the function should return each instance of such numbers in the first list. You can assume that each item in the list is a number and that there are no nested lists. For example, if the function is called in_both_lists:

(in_both_lists '(1 2 3 4 5 7 7 8 9 2 3 4 2 5 9) (2 4 5)) returns *(2 4 5 2 4 2 5)*
(6 marks)

e) Write a tail recursive version of your answer to part d). Make sure both your versions return lists with the elements in the same order.
(8 marks)

PTO

4)

a) Write a function in Lisp that takes a single integer argument, *N*. Your function should return a list of *N* elements – the numbers 1..N. For example, if the function is called NList:
(NList 5) returns (1 2 3 4 5).
(10 marks)

b) Write a tail recursive function in Scheme that accepts a list of numbers and returns a list with every element of the list multiplied by its position in the list. For example if the function is called *listMult*:

(listMult '(2 3 4 5 6)) returns (2 6 12 20 30) – e.g. (2*1 3*2 4*3 5*4 6*5)

(15 marks)

5)

a) Give an example of facts, rules and queries all having the same name and arity. In your own words describe why this can be useful when writing Prolog code.
(5 marks)

b) In your own words describe the Closed World Assumption. How is the Closed World Assumption used in Prolog? What effect does it have on the facts that need to be provided to Prolog programs? What would be required if the Closed World Assumption did not apply?
(6 marks)

c) Write a Prolog rule that finds the sum of all of the odd numbers in a list. You can assume that all of the elements in the list are numbers greater than 0. For example:

?- findSumOdd([1, 2, 3, 4, 5, 6, 7, 8], SumOdd).
SumOdd = 16
(14 marks)

PTO

6)

a) In your own words describe how lists are handled in Prolog. Use examples to illustrate your answer, showing the operations you can perform on lists.
(7 marks)

b) Write code in Prolog that doubles every even number in a list – the odd numbers are left alone. You can assume that all of the elements in the list are numbers. For example:

?-double_even([1, 2, 3, 4, 5, 6, 7], X).
X = [1 4 3 8 5 12 7]
(8 marks)

c) Write Prolog rules which take two lists as arguments. They should count the number of elements in the 2nd list that are equal to the last element in the 1st list. For example:

?-countEqualLast([4,5,6,7,8,9,3], [1,2,3,4,3,5,3,6,7,3,3], Count).
Count = 5
As 3 is the last element in the first list and there are 5 3's in the 2nd list.
(10 marks)

END