Ollscoil na hÉireann, Gaillimh ***GX_____***

National University of Ireland, Galway

Semester I Examinations 2016/ 2017

Exam Code(s)	3BCT , 3BS9
Exam(s)	B.Sc. in Computer Science & Information Technology
Module Code(s)	CT331
Module(s)	Programming Paradigms
Paper No.	1
Repeat Paper	
External Examiner(s)	Dr. John Power
Internal Examiner(s)	Dr. Jim Duggan *Dr. Hugh Melvin *Dr. Frank Glavin

Instructions:

Use separate answer books for each section.
Answer Q1 and one other question from Section A
Answer any two questions from Section B.
All questions carry equal marks.

Duration	2 hours
No. of Pages	5
Department(s)	Information Technology
Course Co-ordinator(s)	Dr. Des Chambers

Requirements:

MCQ
Handout
Statistical Tables
Graph Paper
Log Graph Paper
Other Material

Section A

Answer Q1 and one other question from this section

Q.1 (a) Briefly explain each of the following 4 concepts, describing why they are potentially useful in programming and use code snippets in an appropriate language to illustrate and explain your answer:

(i) Lazy Evaluation
(ii) Closure
(iii) Lambda function
(iv) Currying

(4 x 5)

(b) Regarding the distinction between 1st class functions and higher order functions, an explanation on Stack Overflow states that “ "has first-class functions" is a property of a language, and "is higher-order" is a property of a function”.
Briefly explain this statement.

(5)

Q.2 (a) “The Functional programming paradigm minimise side effects” – explain what this means and why it is important? (5)

(b) Implement a generic swap procedure in C that takes any fundamental data type (int, float, short, string) as arguments. Your `swap( )` prototype should look like

```
void swap(void * vp1,void * vp2, int size)
```

In particular, write C code to show how you would use it to swap two strings, explaining how you would call the `swap` procedure and illustrating what is happening in memory.

(10)

(c)

Write C code to process a student’s grade. It must use 2 separate functions as follows:

- 1st function determines Grade i.e. returns a string. It receives the average of 3 subject results and uses following criteria for Grade:
    - >= 40% : Pass
    - < 40% : Fail
- 2nd function determines average of 3 subject grades (0-100) and returns average as an integer .

You must use function pointers to nest functions. In `main()` read in 3 subject results as floats. The definition of your final procedure called from `main()` should look like,

```
char* Cal_Grade(char*(*fn1)(int),int(*fn2) (float,float,float),
                float sub1,float sub2,float sub3)
{
return fn1(fn2(sub1,sub2,sub3));
}
```

(10)

Q.3 (a) Briefly describe the role of, and relationship between **generators** and **iteriators,** using an example in python to illustrate your answer.
Outline also how these concepts are related to **lazy evaluation.**

(7)

(b) Write a scheme function that determines the minimum value in a list of numbers (You cannot use the built in (`apply min` …) !

(13)

(c) Write a python function that takes a list of numbers and returns those elements in the list that are evenly divisible by 9. You must use **filter** and **lambda** in your answer.

(5)

Section B
Answer any two questions from this section

Q.4

a) With the aid of examples for each, distinguish between *facts*, *relations*, *rules* and *queries* in Prolog.

(8 marks)

b) Describe what is meant by the term *unification*. In your answer, you should outline how constants, structures and variables can be unified in PROLOG.

(5 marks)

c) Describe the main features of a Logic Programming language.
Using an example, explain what is meant by the term *Negation-As-Failure.*
What are the differences between *forward chaining* and *backward chaining?*

(3 x 4 marks)

Q.5

a) Describe, with the aid of examples, the *list* data structure in PROLOG, outlining its representation and syntax. Write code in PROLOG to merge two lists, explaining the steps taken in developing the code.

(8 marks)

b) Explain what is meant by the term *tail recursion*. Write PROLOG code to reverse the items (top level only) in a list, writing **both** a tail recursive and non-tail recursive version of the code. You should explain how each version works.

(9 marks)

c) Describe how the "is" operator works in Prolog. Write Prolog code which returns true if a given list of items is sorted in *descending* order. Otherwise it will return false. For example:

```
sorted([1,2,4,7]).
        false
sorted([4,3,2,1]).
        true
```

(8 marks)

Q.6

a) Describe what is meant by each of the following with respect to program translation:

- Just-In-Time Compilation.
- Lexical Analysis.
- Error Recovery.
- Semantic Analysis.
- Peephole Optimisation.

(5 x 2 marks)

b) Explain what is meant by a Finite State Automaton (FSA). Draw an FSA to recognise strings that contain 'acat' as a substring with alphabet acgt

(8 marks)

c) The following grammar describes a restricted set of assignment expressions that only use addition:

G = {N, T, S, P}
T = { =, +, 0, 1, 2, 3, 4, 5, 6,7 8, 9, a, b}
N = {<R>, <E>,<D>, <id>, <num> }
S = R
P =

```
<R>      ::= <id> = <E>
<E>      ::= <D>| <E> + <D>
<D>      ::= <id> |<num>
<num>    ::= 0|1|2|3|4|5|6|7|8|9
<id>     ::=a|b
```

Are the following strings valid in the above defined grammar?

- "b = 0"
- "a = a + b + 3"

You should create the corresponding parse trees for your validations

(7 marks)