NUI Galway
OÉ Gaillimh

Semester 1 Examinations 2017/ 2018

Course Instance Code(s)	3BCT, 3BS9
Exam(s)	B.Sc. in Computer Science & Information Technology
Module Code(s)	CT331
Module(s)	Programming Paradigms
Paper No.	1
External Examiner(s)	Dr. Jacob Howe
Internal Examiner(s)	*Aidan Breen Dr Michael Schukat

Instructions: Answer 4 questions.
Answer 1 question from section A.
Answer 3 questions from section B.
All questions in section B carry equal marks.

Duration	2 hours
No. of Pages	6
Department(s)	Information Technology
Course Co-ordinator(s)	Dr Des Chambers

PTO

Section A

Answer one of the questions from this section

All questions carry 30 marks

Question 1

A) What is meant by a Programming Paradigm?
(5 marks)

B) With the aid of examples from programming languages of your choice, distinguish between the Procedural and Functional programming paradigms.
(5 marks)

C) In the context of programming paradigms and programming languages, distinguish between Syntax and Semantics.

Using examples from programming languages of your choice, demonstrate a difference in syntax between two languages from different paradigms.
(5 marks)

D) Identify three influences on programming paradigms and briefly describe how each may affect a programming language.
(3 x 5 marks)

Question 2

A) What is meant by Imperative and Procedural programming?
(5 marks)

B) With the aid of examples from programming languages of your choice, distinguish between the *Functional* and *Declarative* programming paradigms.
(5 marks)

C) In the C programming language, what is meant by the term *pointer*?

Describe, using code examples, two common uses of pointers in C.
(2 x 5 marks)

D) What is meant by the term *Side Effect* in relation to procedural programming?

Write a C function that uses pointers to demonstrate a *side effect*, and a line of C code calling that function. You do not need to write an entire C program.
(10 marks)

Section B
Answer three of the questions from this section
All questions carry 25 marks

Question 3

A) Using examples (including code), describe what is meant by the Stack and the Heap in relation to the C programming language.
(4 marks)

B) In relation to the C programming language, answer the following questions using suitable examples:

When handling sensitive data, what extra steps should be taken using the library function free() to delete data?

What is the purpose of the typedef keyword?
(7 marks)

C) Write C code that defines a struct called personStruct with members that store a name as a character array, a birth year as an integer and a description as a pointer to a character array.

Write a C function called deletePerson that accepts a pointer to a personStruct instance and frees all of the memory associated with that struct.
(14 marks)

Question 4

A) Using examples, describe what is meant by the type modifiers *short, long, signed* and *unsigned* in relation to the C programming language.
(4 marks)

B) In relation to the C programming language, answer the following questions using suitable examples:

What is a struct?
What is the function of sizeof()?
(7 marks)

C) 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.
(14 marks)

Question 5

A) Distinguish between the Scheme primitives car and cdr by writing sequences of car and cdr to extract the item "twenty" from the following lists:

```
(this week her mother bought twenty cupcakes)

((this week her) mother (bought twenty cupcakes))

((this week her) (bought ((twenty)) cupcakes))
```

(4 marks)

B) Write a recursive function in Scheme which returns a list of all numbers lower than some value when passed a list of numbers and that value. You may assume each item in the list is a number and there are no nested lists. For example, if the function is called nums_below:

```
(nums_below '(1 2 3 4 5) 3) returns (1 2)

(nums_below '(1 2 3 4 5) 2) returns (1)
```

(7 marks)

C) Write both a non-tail recursive and a tail recursive function in Scheme which calculates the sum of all numbers in a list. You may assume each item in the list is a number and there are no nested lists. For example, if the function is called get_sum:

```
(get_sum '(5 5 5 1)) returns 16

(get_sum '(1 2 3 4 5 6)) returns 21
```

Explain the approach taken highlighting the base case and reduction stages for each function.
(14 marks)

Question 6

A) "The Functional programming paradigm aims to minimize side-effects." Explain why minimizing side-effects might be desirable and the limitations of strict side-effect free code.
(4 marks)

B) Using suitable examples and code, explain what is meant by a *well defined recursive function*. What are the benefits of a well defined recursive function? What are the drawbacks of a recursive function that is not well defined?
(7 marks)

C) Write a tail recursive function in Scheme that accepts a list of numbers, and returns a list of two numbers. The first number in the returned list should contain the sum of all even numbers in the input list. The second number in the returned list should contain the sum of all odd numbers in the input list. For example, if the function is called split_sum:

```
(split_sum '(1 2 3 4)) returns (6 4)

(split_sum '(1 1 1 4 4)) returns (8 3)
```

How could split_sum be changed to become a higher-order function, so that the input list can be split up in some way other than by even and odd numbers based on an input function reference?
(14 marks)

Question 7

A) With the aid of examples for each, distinguish between *facts*, *rules* and *queries* in Prolog.
(4 marks)

B) Describe what is meant by the term *unification.* In your answer, you should provide an example set of facts, rules and queries to demonstrate unification.
(7 marks)

C) Write a set of prolog facts and rules that count the number of times a given element appears in a given list of numbers. For example, if the rules were named 'countInstances':

```
?- countInstances([1, 2, 3, 3, 4, 3, 5], 3, X).
X=3.
```

(14 marks)

Question 8

A) Describe, with the aid of examples, the *list* data structure in Prolog, outlining its representation and syntax.
(5 marks)

B) Write code in Prolog to merge two lists, explaining the steps taken in developing the code.
(10 marks)

C) 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.
```

(10 marks)

END