جامعة بيروت العربية
BEIRUT ARAB UNIVERSITY

Date: 20-12-2022

Duration: 1.5 Hrs

Mark: / 40

No. of pages including cover: 8

Final Exam

Semester: Fall 2022/2023

Faculty	: Faculty of Science		
Department	: Math & Computer Science	**Division/ program**: Computer Science	
Course Name	: Concepts of Programming Languages	**Course Code**: CMPS 445	

Student's Name: [illegible]	**ID:** ....
Section/ Group:	**Seat Number:** ....

INSTRUCTIONS:

1- Any kind of cheating will subject the student to the penalties specified by the University rules

2- Use of cell phone is strictly prohibited

Question	Mark	Out of
One		**15**
Two		**4**
Three		**8**
Four		**4**
Five		**4**
Six		**5**
Total		**40**

Total marks in letters

Examiner's Name: Dr May Itani

Signature: ....

<u>Question 1. MCQs + SA (15 pts)</u>

Check if the given statement is ***True*** or ***False*** and circle the relevant choice**.** If ***False*** correct the statement (**without negating** the statement) and write it in the provided space below (empty row).

Garbage collection is a solution for the dangling pointer problem	**T F**
LISP is often cited as an example of an imperative programming language	**T F**
Associativity rules only apply to operators of the same precedence level	**T F**
The execution of ***Coroutines*** is interleaved, but not overlapped	**T F**
One disadvantage of ***Pass-by-reference*** is that it requires extra storage location and copy operation	**T F**
The Scheme interpreter is an infinite read-evaluate-print loop	**T F**
Statements are the fundamental means of specifying computations in a programming language	**T F**
Logically-Controlled Loops provide selection control based on a Boolean expression	**T F**

<u>MCQs</u>

1. ---------------- conversion is one that converts an object to a type that cannot include all of the values of the original type

   a) narrowing
   b) widening
   c) mixed mode
   d) coercion

2. **`(double)a`** where a was originally declared as **`int`** is considered

   a) Explicit widening conversion
   b) Implicit widening conversion
   c) Explicit narrowing conversion
   d) Implicit narrowing conversion

3. In many programming languages, ‘otherwise’ and ‘else’ are part of which building block?

   a) iteration
   b) counter
   c) selection
   d) list structure

4. What distinguishes a purely “functional” programming language from an “imperative” one?

   a) There are no variables and hence no assignment operation in a purely functional language
   b) A purely functional language lacks the ``go to'' statement, but an imperative language always has such a command
   c) All subprograms must be declared with the keyword function in a purely functional language
   d) There is no real difference, only a difference in the recommended coding style

5. In which of the following expressions short circuit evaluation maybe applied (Assume we are using C, C++ or Java)?

   a) `if(2 + 2 == 4 || 0 == 1)…`
   b) `if(2 + 2 == 4 | 0 == 1)…`
   c) Both a & b
   d) None of the above

6. The primary activity of a prolog interpreter is

   a) Unification
   b) Instantiation
   c) Resolution
   d) None of the above

7. Which of the following is considered a deficiency in prolog?

   a) Inference engine
   b) Resolution order
   c) Forward chaining
   d) None of the above

Question 2 (4 pts)

Part A.

Assume a programming language uses the following rules of associativity and precedence for expressions:

Precedence	*Highest*	*, /, **not**
		+, −, &, **mod**
		- (unary)
		=, /=, <, <=, >=, >
		and
	Lowest	**or**, **xor**
Associativity	*Left to right*	

Show the order of evaluation of the following in details. Show all steps. Give intermediate values then the final value of the expression based on the given precedence and associativity rules given above.

Expression 1
```
a * b - 1 + c
```

Expression 2
```
(a - c) / b & (d * e / a - 10)
```

<u>Part B.</u>
Show the order of evaluation of the expressions of Part A, assuming that there are **no precedence rules** and all operators associate right to left.

Expression 1
```
a * b - 1 + c
```

Expression 2
```
(a - c) / b & (d * e / a - 10)
```

Question 3. (8 pts)

Consider the following C-like skeletal code:

```
int a[ ] = { 0, 2, 1, 2 };
void guessWhat(int x, int y) {
        x = x + 1;
        a[x] = a[y] + 1;
        y = y + 1;
}
int b = 1;
guessWhat(b, a[b]);
```

What are the values of variables **a** and **b** after the call to the subprogram **guessWhat** returns when:

(a) Pass-by-value is used,

(b) Pass-by-result is used,

(c) Pass-by-value-result is used,

(d) Pass-by-reference is used, and

(e) Pass-by-name is used?

Assume that the language uses 0-based indices for arrays and an uninitialized variable is set to a default value, e.g., 0 for ***int*** variables.

Show your work

Method	Variable **a** (all elements in the array)	Variable **b**
Pass-by-value		
Pass-by-result		
Pass-by-value-result		
Pass-by-reference		
Pass-by-name		

Question 4 (4 pts)
Functional Programming (Scheme)

A. In the following table, the first column lists some s-expressions containing a single instance of the atom X. Assume that the variable S is bound to the s-expression in the first column. The second column should be an expression that when evaluated **returns the atom X**. You are only allowed to use the functions CAR and CDR and a single instance of the variable S. The first two rows show sample answers.

s	**Expression to return atom x**
(a b x)	***(car (cdr (cdr s)))***
(((x)))	***(car (car (car s)))***
(a (b (c x)))	
(a (b (c)) x)	

B. What does the given scheme function do?

```
(define (mystery s n) (if (= n 0) s (mystery (+ s n)
(- n 1))))
```

Explain then give an example to justify your answer.

Question 5 (4 pts)
Logic Programming (Prolog)

```
father(jack, susan).
father(jack, ray).
father(david, liza).
father(david, john).
father(john, peter).
father(john, mary).
mother(karen, susan).
mother(karen, ray).
mother(amy, liza).
mother(amy, john).
mother(susan, peter).
mother(susan, mary).
```

```
parent(X, Y) :- father(X, Y).
parent(X, Y) :- mother(X, Y).
grandfather(X, Y) :- father(X, Z), parent(Z, Y).
grandmother(X, Y) :- mother(X, Z), parent(Z, Y).
grandparent(X, Y) :- parent(X, Z), parent(Z, Y).
yeye(X, Y) :- father(X, Z), father(Z, Y).
mama(X, Y) :- mother(X, Z), father(Z, Y).
gunggung(X, Y) :- father(X, Z), mother(Z, Y).
popo(X, Y) :- mother(X, Z), mother(Z, Y).
```

How many facts, rules, predicates are there in the knowledge base?

Answer the following queries

```
?- parent(susan, mary).
```

```
?- parent(ray, peter).
```

```
?- mama(amy, X).
```

```
?- gunggung(X, Y).
```

Question 6 (5 pts)
Implementing Subprograms

Show the stack with all activation record instances, including static and dynamic chains, when execution reaches position 1 in the following skeletal program. This program uses the deep-access method to implement dynamic scoping.

```
void fun1() {
float a;
. . .
}
void fun2() {
int b, c;
. . . <--------- 1
}
void fun3() {
float d;
. . .
}
void main() {
char e, f, g;
. . .
}
```

The calling sequence for this program for execution to reach `fun3` is

`main` calls `fun1`
`fun1` calls `fun3`
`fun3` calls `fun2`