Nalongsone Danddank Student ID : 14958950 StarID: jf3893pd

Email: [nalongsone.danddank@my.metrostate.edu](mailto:nalongsone.danddank@my.metrostate.edu)\

**Metropolitan State University**

**ICS-365 Exam #1**

**75 Points**

1. **Make sure you read the questions carefully and answer each part.**
2. Some questions may be answered with as little as 2 sentences. Some may require more. No question should require more than 6 sentences, but use whatever space you feel necessary.
3. Upon completion of exam, make sure to upload it to D2L.
4. You will be graded on completeness and quality of your answers. So it is possible to have an answer with the correct information, but not be complete.
5. Use the Word Highlighter function to answer the Multiple Choice questions.

**Multiple Choice (10x1 = 10 points)**

1. What generally produces faster execution
   1. None of the below
   2. A purely interpreted language
   3. A hybrid language
   4. A purely compiled language
2. Select the correct order in the compilation process.
   1. Syntax Analysis then Lexical Analysis
   2. Code Generation, then Syntax, then lexical analysis
   3. Lexical Analysis, then Syntax Analysis, then Code Generation
   4. None of the above
3. In what language is Unix mostly written?
   1. ALOGOL
   2. Assembly
   3. ‘C’
   4. ‘C++’
4. Attribute grammars cannot capture and/or enforce some features in a language that BNF grammars cannot.
   1. TRUE
   2. FALSE
5. If the grammar for a language is ambiguous, then some valid expression in that language has more than one parse tree.
   1. TRUE
   2. FALSE
6. ALGOL 60 was defined using which meta-language. (Choose the best answer)
   1. BNF
   2. EBNF
   3. XML
   4. UML
7. What was wrong with using machine code. (Choose the best answer)
   1. Poor readability
   2. Poor modifiabiliy
   3. a, b and d
   4. Expression coding was tedious
8. What are Predicate Functions?
   1. They are only applicable in human languages, not computer languages.
   2. They state the static semantic rules for a language and are associated with grammar rules.
   3. They state the grammar rules of a language and are associated with semantic rules.
   4. There is no such thing as a predicate function.
9. What is the difference between synthesized and inherited attributes?
   1. Inherited attributes depend on information from descendants in the parse tree.
   2. Inherited attributes are useful only in object-oriented languages.
   3. Synthesized attributes are not computed.
   4. Synthesized attributes depend on information from descendants in the parse tree.
10. Which is not true about Derivations?
    1. A derivation must be either leftmost or rightmost
    2. A sentence is a sentential form that has only terminal symbols
    3. A leftmost derivation is one in which the leftmost nonterminal in each sentential form is the one that is expanded
    4. Every string of symbols in a derivation is a sentential form

**Pseudocode (choose 1, 5 points)**

1. Pseudocodes were invented to solve many problems. What were two principle issues designers where trying to overcome?

- It is easier for people to understand than conventional programming language code and to ensure that programmers understand a project’s requirements.

- it is an efficient and environment-independent description of the key priciples of an algorithm.

1. The Pseudocode is described by four principles. (automatation, orthogonality, regularity, security) Give a description of 3 of these principles)

Automation: In this principle the code is said to have to be automated. Automation means the program does tasks by itself once started. It do not need any human help in between.

**Chapter 1 (10 points, choose 3)**

1. Describe the four categories of programming languages. Give the name and a brief description

Imperative: is a programming paradigm that uses statements that change a program’s state. For instance: Fortran, Algol, C, ...

Functional: is a way of thinking about software construction by creating pure functions.

For instance: Scala, Haskell, Scheme,F#, …

Logic: is languages based on symbolic logic.

For instance: Prolong, Fril, …

Object oriented: is the Languages that support object-oriented programming, the primary design issues for inheritance and dynamic binding.

For instance: C++, Object-C, Java, C#…

1. What does orthogonality mean?

a relatively small set of primitive constructs can be combined in a relatively small number of ways to build the control and data structures of the language; every possible combination of primitives is legal and meaningful.

1. Name and describe 3 of the 5 programming domains listed in the book.

Scientific Applications: were invented and used for scientific applications. In that time used relatively simple data structures, but required large numbers of floating-point arithmetic computations. Use of arrays. The most common used is: Fortran.

Business Applications: for business applications, are characterized by facilities ofr producing elaborate reports, precise ways of describing and storing decimal numbers and character data, and the ability to specify decimal arithmetic operations. The popular used is: COBOL.

Artificial Intelligence: is area of computer applications characterized by the use of symbolic rather than numeric computations. Use of linked lists. The popular used is: LISP.

Web Software: is supported by an eclectic collection of languages, ranging form markup language, HTML, which not a programming language. And scripting like Javascript, PHP, general-pupose like Java.

1. There are 4 main criteria for language evaluation, choose and describe 2 of them.

***Readability***: the ease with which programs can be read and understood

- Overall simplicity: a manageable set of features and constructs. Minimal feature multiplicity(number of ways to do the same thing). Minimal operator overloading (number of meanings a single operator symbol has).

- Orthogonality : A relatively small set of primitive constructs can be combined in a relatively small number of ways; Every possible combination is legal.

- Data types: Adequate predefined data types,Ex: having Boolean instead of just 0 and 1.

- Syntax considerations: Special words and methods of forming compound statements

Easier to read if the language uses end-if or end-loop, rather then just a bracket.

Form and meaning: self-descriptive constructs, meaningful keywords The reserved word “static” can mean different things in C, depending on where it is used.

***Writability***: Considered within the domain of the language, Visual Basic good for GUI, and C for operating systems. Comparing the two directly is not really fair.

Simplicity and orthogonality: Few constructs, a small number of primitives, a small set of rules for combining them, but yet enough to be useful (balance)

Support for abstraction: The ability to define and use complex structures or operations in ways that allow details to be ignored

Expressivity: A set of relatively convenient ways of specifying operationscount++

Strength and number of operators and predefined functions

**Chapter 2 (5 points)**

1. Algol 68 was special for several reasons. One can easily think of four, but I would like you describe 2 reasons why it was special. You can site reasons like implementation details, features, acceptance, etc. (but you need to describe them).

Algol 68 stands for Algorithmic Language 1968. and base on the late - 1968 draft Report, was introduced by the Royal Radar Establishment in the UK as ALGOL 68-R in July 1970.

Its special features can be listed to: It was imperative and a widely use

Acceptance: became a widely-used language in military coding, especially in the UK

**Chapter 3 (25 Points)**

1. Consider the grammar given below (5 points)

<pop> ::= [ <bop> , <pop> ] | <bop>

<bop> ::= <boop> | ( <bop> )

<boop> ::= x | y | z

(a) What are the nonterminal symbols? (1 point)

<bop>, <pop> , <boop>

(b) What are the terminal symbols? (1 point)

x, y, z

(d) Draw a parse tree for the sentence (x). (3 points)

(note the parenthesis are part of the sentence)

<pop>

|

<bop>

|

(<bop>)

|

(<boop>)

|

(x)

1. (10 points) Show that the following grammar is ambiguous (Hint: first two levels

of both trees will be the first line)

<S> -> <A>

<A> -> <A> + <A> | <id>

<id> -> a | b | c

ambiguous #1: a + b + c

<S>

|

<A>

/ | \

<A> + <A>

| / | \

<id> <A> + <A>

| | |

a <id> <id>

| |

b c

ambiguous #2: a + b + c + c

<S>

|

<A>

/ | \

<A> + <A>

/ | \ / | \

<A> + <A> <A> + <A>

| | | |

<id> <id> <id> <id>

| | | |

a b c c

1. (10 points) Using the following grammar, show the leftmost derivation for:

A = A \* (B + (C \* A)) (derivation, not the parse tree)

<assign> → <id> = <expr>

<id> → A | B | C

<expr> → <id> + <expr>

| <id> \* <expr>

| ( <expr> )

| <id>

Leftmost derivation:

<assign> -> <id> = <expr>

A = <expr>

A = <id> \* <expr>

A = A \* <expr>

A = A \* (<expr>)

A = A \*(<id> + <expr>)

A = A \* (B + <expr>)

A = A \* (B + (<expr>))

A = A \* (B + (<id> \* <expr>))

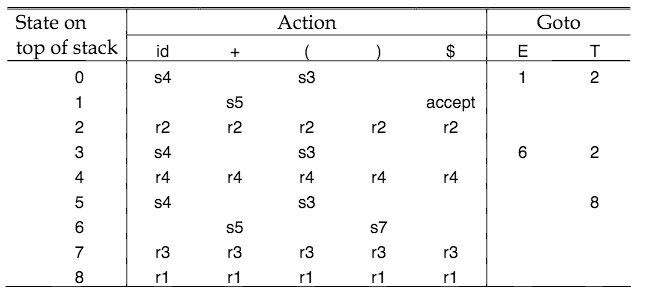
A = A \* (B + (C \* <expr>))

A = A \* (B + (C \* <id>))

A = A \* (B + (C \*A))

1. **Chapter 4 (10 Points)**

Consider the following state table:



Here is the associated grammar:

1) E –> E + T

2) E –> T

3) T –> (E)

4) T –> id

Show the parse of (id) + (id) using the LR algorithm (Hint: Shift-Reduce Algorithm) practiced in class.

Solution:

Stack Inout Action

1. (id) + (id)$ s3

0(3 id) + (id)$ s4

0(3id4 ) + (id)$ r4[3, T]

0(3T2 ) + (id)$ r2[3, E]

0(3E6 ) + (id)$ s7

0(3E6)7 + (id)$ r3[0, T]

0T2 + (id)$ r2[0, E]

0E1 + (id)$ s5

0E1+5 (id)$ s3

0E1+5(3 id)$ s4

0E1+5(3id4 )$ r4[3, T]

0E1+5(3T2 )$ r2[3, E]

0E1+5(3E6 )$ s7

0E1+5(3E6)7 $ r3[5, T]

0E1+5T8 $ r1[0, E]

0E1 $ accept

it is **accepted** in the grammar.

1. **C Programming (10 Points) – write a C program that has a main function and one other function. The program will accept three different positive integer numbers from the user, and print out the smallest number. The main function will handle the input and the output, the other function will just do the determination of the smallest number and return the results. Assume all three numbers are positive integers, and all three are different numbers (no edit checking required).**

/\* @Author: Nalongsone Danddank Student ID :14958950 StarID: jf3893pd

\* Email: nalongsone.danddank@my.metrostate.edu\

\* ICS 365. Organization of Programming Languages.

\* Exam #1

\*/

#include <stdio.h>

#include <stdlib.h>

// declaration function.

int smallest\_num(int, int, int);

int main(int argc, char \*argv[]) {

// initilize valiables for user input

int x,y,z;

//show infomation to user to input the 3 number.

printf("Input the 3 three different positive integer numbers:\n");

printf("Input the first positive integer:");

scanf("%d", &x);

printf("Input the second positive integer:");

scanf("%d", &y);

printf("Input the thirst positive integer:");

scanf("%d", &z);

//display the result.

printf("Minimum of 3 numbers that you input is %d\n", smallest\_num(x, y, z));

printf("Done! Bye!");

return 0;

}

/\* for calculating the minimum of the 3 numbers.

\* @params a - int

\* b - int

\* c - int

\* @return smallest of 3 number - int.

\*/

int smallest\_num(int a, int b, int c){

if (!(b / a))

return (!(b / c)) ? b : c;

return (!(a / c)) ? a : c;

}

Show the result:

