National University of Computer and Emerging Sciences

Software Re-engineering (SE4001)

Quiz-III –(BSE-8B)

Date: April 24th 2024

Course Instructor(s)

Dr. Syed Muazzam Ali Shah

Total Time: 30 minutes

Total Marks: 10

Total Questions: 02

Semester: SP-2024

Campus: Karachi

Dept: Software Engineering

________________________________ ________ ________ ________________________

Student Name Roll No Section Student Signature

Q 1: Select the most appropriate option from the following multiple-choice questions. (5 marks)

1. Which one of the following is among the six reverse engineering objectives as identified by Chikofsky and Cross II:

a. Refactoring
b. Data re-engineering
c. Restructuring
d. Reusability

2. In a programming language, the parser might classify keywords, identifiers, operators, and punctuation symbols as separate token types. **[True/False]**

3. Reverse engineering is performed to achieve two objectives.

a. Documentation of Artifacts and Design recovery
b. Refactoring and data re-engineering
c. Design recovery and Refactoring
d. Data re-engineering and documentation of artifacts

4. High level reverse engineering means to create abstractions of source code in the form of design, architecture and/or documentation. **[True/False]**

5. Which one of the following is not considered as a token category in lexical analysis?

a. Keywords
b. identifiers
c. literals
d. pre-processor directives

Q 2: **Using lexical analysis generate all possible valid tokens, specific their categories, and also specify the number of total valid token using the following piece of source code.** ***(5 marks)***

```
#include <stdio.h>
#include <iostream.h>
Int main () {
   Int a = 10;
    if (a < 2) {
      printf (“a is less than 2”);
   } else {
      printf (“a is not less than 2”);
   }
   printf (“value of a is: %d\n”, a);
   return 0;
}
```

Sr#	**Lexeme**	**Token**
1	Int	keyword
2	main	identifier
3	(	operator
4	)	operator
5	{	operator
6	Int	keyword
7	a	identifier
8	=	operator
9	10	identifier
10	;	separator
11	if	keyword
12	(	operator
13	a	identifier
14	<	operator
15	2	identifier
16	)	operator
17	{	operator
18	printf	keyword
19	(	operator
20	“a is less than 2”	string
21	)	operator
22	;	separator
23	}	operator
24	else	keyword
25	{	operator

26	printf	keyword
27	(	operator
28	“a is not less than 2”	string
29	)	operator
30	;	separator
31	}	operator
32	printf	keyword
33	(	operator
34	“value of a is: %d\n”	string
35	,	separator
36	a	identifier
37	)	operator
38	;	separator
39	return	keyword
40	0	constant
41	;	separator
42	}	operator