

CSE 310 Quiz (Section A)

* Required

1. Email *

2. Name (maximum 50 characters) *

3. Write your 7-digit student number below, e.g., 1705001. *

4. Mobile No. (11-digit) *

Instructions and Pledge

5. I hereby declare that, I shall not misuse, in any form or method, the course materials including but not limited to, Lecture Notes, Reading Materials, Audio and Video of Lectures of this course, Codes and Editors. I shall not adopt any unfair means during the Quiz exam and shall not receive any help or offer/provide help to anyone in any manner whatsoever. I will not expose the hard copy and soft copies of the questions and answers to any person/party/media. I agree to accept any punitive measure taken by the BUET Authority, if at any time during or after completion of the course if it is revealed/violated otherwise. *

Check all that apply.

☐ I agree

6. Read the below instructions carefully. *

Instructions

1. Total marks is 40. All questions are *not* of equal marks. Please find the marks allotted for each question to its right.
2. There will be negative marking. 25% marks of a question will be deducted for an incorrect answer.
3. Total time is 30 minutes. No submission will be accepted after the time is over.
4. Close all resources such as books, slides, codes, other tabs in your browser etc.
5. Shut down all other communication media that are not being used for taking the quiz.
6. Periodically save your answers by pressing the Submit button. You can edit your responses as many times you want within the exam duration.
7. Keep your camera TURNED ON all the time.

Check all that apply.

☐ I have read all the instructions.

Password

Password Will be provided by the instructors.

7. Password *

Questions

25% marks of a question will be deducted for an incorrect answer.

8. In the symbol table you constructed, when we need to get the information of a symbol, at first: 1 point

Mark only one oval.

- ☒ we will search at the topmost scope-table.
- ☐ we will search at the bottom-most scope-table.
- ☐ we will search at a random scope-table.
- ☐ we will search the nearest scope-table.

9. In your first assignment, chaining was used:

1 point

Mark only one oval.

- ☒ for avoiding collisions in hash tables.
- ☐ for a better binding between the name and type of a symbol.
- ☐ for printing detailed information of names.
- ☐ for printing detailed information of types.

10. Which of the following may be termed as lexical errors? i. Unfinished string ii. Unfinished comment iii. Unrecognized character iv. character sequence like 1.2.345 1 point

Mark only one oval.

- ☒ All of i., ii., iii., and iv.
- ☐ Only iv.
- ☐ iii. and iv.
- ☐ None of i., ii., iii., and iv.

11. Which of the following is the correct sequence of sections in a Flex file?

1 point

Mark only one oval.

- ☒ definitions, %, rules, %, user code
- ☐ definitions, %, user code %, rules
- ☐ definitions, %, %, rules, user code
- ☐ definitions, %, rules, user code, %

12. In Flex, $([0-9])^+ \cdot ([0-9])^*$ will match:

1 point

Mark only one oval.

☒ 12.3

☐ 12a3

☐ Both of 12.3 and 12a3

☐ None of 12.3 and 12a3

13. Which of the following files is by default generated in Flex?

1 point

Mark only one oval.

☒ lex.yy.c

☐ flex.yy.c

☐ lex.y.c

☐ [lex.yy.cc](#)

14. Which of the following was the Flex library used by you?

1 point

Mark only one oval.

☒ libfl.a

☐ libfl.y

☐ libfa.l

☐ flibl.a

15. In Flex, which of the following holds the text of the current token?

1 point

Mark only one oval.

☒ char *yytext

☐ char yytext

☐ char *yyin

☐ char **yytext

16. Which of the following is the file which by default Flex reads from?

1 point

Mark only one oval.

☒ FILE *yyin

☐ FILE yyin

☐ FILE *yin

☐ FILE *yyyin

17. To interface with a bison (yacc) file, which of the following needsto be put inside the Flex file?

1 point

Mark only one oval.

☒ %{ #include "y.tab.h" %}

☐ { #include "y.tab.h" }

☐ %%{ #include "y.tab.h" %%}

☐ % #include "y.tab.h" %

18. Consider the grammar you used in your YACC offline. Which of the following rules would be left unused in parsing the following code snippet: `int func (int a, int b){return a+b;}` 1 point

Mark only one oval.

- ☐ `compound_statement : LCURL statements RCURL`
- ☒ `statement : var_declaration`
- ☐ `logic_expression : rel_expression`
- ☐ `type_specifier : INT`

19. Consider the YACC code snippet below. According to the code, how the input `2#3#4^5^6` would be grouped? 2 points

```
%left '#'
%right '^'
```

```
%%
```

```
...
```

```
exp: exp '#' exp
    | exp '^' exp
    | NUMBER
    ;
```

Mark only one oval.

- ☒ `(2#3)#(4^(5^6))`
- ☐ `2#(3#(4^(5^6)))`
- ☐ `((((2#3)#4)^5)^6)`
- ☐ `(2#3)#4^(5^6)`

20. The semantic error “void function used in expression” has to be checked in which of the following rule’s corresponding action code? 1 point

Mark only one oval.

- ☐ func_definition : type_specifier ID LPAREN parameter_list RPAREN
compound_statement
- ☐ statement : RETURN expression SEMICOLON
- ☒ simple_expression : simple_expression ADDOP term
- ☐ All of the mentioned rules

21. How many tokens do a YACC/bison parser look ahead while parsing? 1 point

Mark only one oval.

- ☐ None
- ☒ 1
- ☐ 2
- ☐ 3

22. You want to catch the semantic error “function definition not in global scope”. For this you intend to write an extra rule to catch the semantic error without introducing any conflicts. Which of the following rule should you write? 1 point

Mark only one oval.

- ☐ This error cannot be handled by writing extra rules
- ☒ statement : func_definition
- ☐ compound_statement : LCURL statements dummy RCURL ; dummy : func_definition
- ☐ logic_expression : logic_expression error

23. What is the default definition of YYSTYPE?

1 point

Mark only one oval.

☒ int

☐ string

☐ char *

☐ It is not defined by default

24. In the action code of the rule “comp_exp : simp_exp OP simp_exp”, You want to check whether the two operands of the OP operator are of the same type or not. Which of the following value references must you check? 1 point

Mark only one oval.

☐ \$0 and \$2

☐ \$\$, \$1 and \$3

☒ \$1 and \$3

☐ \$\$ and \$1

25. Which of the following declaration can be used to assign types to tokens?

1 point

Mark only one oval.

☐ %type

☒ %token

☐ %union

☐ Any one of %type or %token can be used

26.

1 point

Consider the following C code snippet:

```
int main() {  
    int a,b,c;  
  
    a = b+c;  
    return 0;  
}
```

Also consider the following three lines of code to be placed in the blank box shown:

- i) //This is a comment
- ii) char str[50] = "This is a string";
- iii) c == func(a);

Placing which of the three lines would result in a syntax error and cannot be fixed without writing extra rules? Consider the grammar provided to you in the syntax & semantics error checking assignment. Also consider that the rest of the code without the extra line would be successfully parsed without any errors.

Mark only one oval.

- ☐ i and ii
- ☒ only ii
- ☐ i and iii
- ☐ i, ii and iii

27. Consider the following three statements. Which of them is true?

1 point

- a. If one does not handle the precedence/ordering of if/else statements, it would lead to a segmentation fault whenever if/else statements occur in the input.
- b. If one has a rule where type of \$\$ does not match the type of \$1, and nothing is written on the corresponding action code of the rule, it will lead to a segmentation fault whenever the rule would be visited during the parsing of the input.
- c. If one has a rule where type of \$\$ does not match the type of \$2, and nothing is written on the corresponding action code of the rule, it will lead to a segmentation fault whenever the rule would be visited during the parsing of the input.

Mark only one oval.

☐ a.

☐ b.

☐ c.

☒ All are false.

28. Which of the following assembly code fragment is used to initialize the data segment register?

1 point

A. `MOV DS, @DATA`
`MOV AX, DS`

C. `MOV @DATA, AX`
`MOV DS, AX`

B. `MOV AX, @DATA`
`MOV DS, AX`

D. `MOV DS, AX`
`MOV AX, @DATA`

Mark only one oval.

☐ A

☒ B

☐ C

☐ D

29. Consider the assembly file(s) that you generated in assignment 4 (ICG). Each file contained multiple procedures such as main, println, foo, bar etc. as determined by the input. However, when the assembly file was compiled and emulated, execution always started from the procedure named 'main'. How did the emulator determine this? 1 point

Mark only one oval.

- ☐ The emulator always executes the first procedure, i.e., the topmost procedure in the .CODE section. Hence the 'main' procedure was deliberately placed at the top during code generation.
- ☐ The emulator expects that there will always be a procedure named 'main' in the assembly file and starts execution from that main procedure (the way a C compiler expects a main function).
- ☐ The emulator starts execution from the procedure which contains the data segment initialization.
- ☒ None of the other three answers

30. Find below a code snippet in 'C' followed by the entries created in the .DATA section of the generated assembly file by different implementations of ICG. Which implementation should be considered (the most) correct? 1 point

```
int a,b[3];  
int f(int x, int y);  
int main() {  
    return 0;  
}
```

A. `a_0 DW ?`
`b_0 DW 3 DUP (0)`

C. `a_0 DW ?`
`b_0 DW 3 DUP (?)`
`x_1 DW ?`
`y_1 DW ?`

B. `a_0 DW ?`
`b_0 DW 3`

D. Both A and B

Mark only one oval.

☒ A

☐ B

☐ C

☐ D

31. Which of the following assembly code fragments represents a correct translation of the statement `a[1]=a[5]` where 'a' is an array of integers of size 10. Assume integers are two bytes long and the target machine is byte addressable. 2 points

A. `MOV t1, 1`
`MOV BX, t1`
`ADD BX, BX`
`MOV t2, 5`
`MOV DI, t2`
`ADD DI, DI`
`MOV AX, a[DI]`
`MOV a[BX], AX`

C. `MOV t1, 1`
`MOV t2, 5`
`MOV BX, t1`
`MOV DI, t2`
`MOV AX, a_1[DI]`
`MOV a_1[BX], AX`

B. `MOV BX, 1`
`ADD BX, BX`
`MOV AX, a[BX]`
`MOV t0, AX`
`MOV BX, 5`
`ADD BX, BX`
`MOV AX, a[BX]`
`MOV t0, AX`

D. None of the above

Mark only one oval.

- ☒ A
- ☐ B
- ☐ C
- ☐ D

32. Find below a code fragment written in 'C' and the corresponding assembly code generated by an implementation of ICG. For convenience, the assembly code has been spread into two columns; read the code in the order from top to bottom and left to right. Now, identify the correct jump labels for each of the blanks marked as A, B, C and D.

4 points

Code Fragment in C

```
if(x>=y)
    if (x>y)
        flag=1;
    else flag=0;
else flag=-1;
```

Generated Assembly Code

```

mov ax, x
cmp ax, y
jle L_0
mov t0, 1
jmp L_1
L_0:
mov t0, 0
L_1:
mov ax, t0
cmp ax, 0
je A

mov ax, x
cmp ax, y
jle L_2
mov t1, 1
jmp L_3
L_2:
mov t1, 0

```

```

L_3:
mov ax, t1
cmp ax, 0
je B

mov ax, 1
mov flag, ax
jmp C
L_4:
mov ax, 0
mov flag, ax
jmp D
L_6:
mov ax, 1
neg ax
mov flag, ax
L_7:

```

Mark only one oval per row.

	L_3	L_4	L_6	L_7
A	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
B	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
C	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
D	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>

33. Find below a code fragment written in 'C' and the corresponding assembly code generated by different implementations of ICG. Which implementation should be considered correct? 2 points

```
k=5;  
while(k>0){  
    k--;  
}
```

A. L_1:
 mov ax, 5
 mov k, ax
L_2:
 mov ax, k
 cmp ax, 0
 jle L_3

 mov ax, k
 dec k
 jmp L_2
L_3:

B. L_1:
 mov ax, 5
 mov k, ax
L_2:
 mov ax, k
 cmp ax, 0
 jle L_3

 mov ax, k
 jmp L_2
L_3:

C. L_1:
 mov ax, 5
 mov k, ax
L_2:
 mov ax, k
 cmp ax, 0
 jle L_3

 mov ax, k
 dec k
 jmp L_1
L_3:

D. None of the above

Mark only one oval.

☒ A

☐ B

☐ C

☐ D

34. Find below a function written in 'C' followed by the assembly codes generated for the function by implementations of ICG from four different students. Which implementation should be considered correct? Assume that the input parameters of the function are pushed into the stack immediately before invoking the function and in the order they are specified in the function definition. The return value is passed through the register AX. Also the system state is saved by the caller before calling and restored by the caller after returning from the function. 2 points

```
int subtract(int a, int b){  
    return a-b;  
}
```

A.

```
subtract proc  
    push bp  
    mov bp, sp  
    mov ax, word ptr [bp + 4]  
    sub ax, word ptr [bp + 6]  
    ret  
subtract endp
```

C.

```
subtract proc  
    push bp  
    mov bp, sp  
    mov ax, word ptr [bp - 6]  
    sub ax, word ptr [bp - 4]  
    ret  
subtract endp
```

B.

```
subtract proc  
    push bp  
    mov bp, sp  
    mov ax, word ptr [bp + 6]  
    sub ax, word ptr [bp + 4]  
    ret  
subtract endp
```

D.

```
subtract proc  
    push bp  
    mov bp, sp  
    mov ax, word ptr [bp - 2]  
    sub ax, word ptr [bp - 4]  
    ret  
subtract endp
```

Mark only one oval.

☐ A

☒ B

☐ C

☐ D

35. Find below an expression written in 'C' and its corresponding assembly code (without any optimization) generated by an implementation of ICG. For convenience the assembly code has been spread into two columns; read the code in the order from top to bottom and left to right. Note that the assembly code uses seven temporary variables. Now, after optimization, what is minimum number of temporary variables that would suffice to produce the same output. Assume that the optimization algorithm only involves reuse of temporary variables and does not consider any other techniques such as use of other registers instead of temp variables or pushing values to stack to save results of subexpressions, etc.

`i = 2*(3+4)*(8-6)+(5+4)*(6+7);`

```
;3+4
MOV AX, 3
ADD AX, 4
MOV t0, AX
```

```
;2*(3+4)
MOV AX, 2
MOV BX, t0
MUL BX
MOV t1, AX
```

```
;8-6
MOV AX, 8
SUB AX, 6
MOV t2, AX
```

```
2*(3+4)*(8-6)
MOV AX, t1
MOV BX, t2
MUL BX
MOV t3, AX
```

```
;5+4
MOV AX, 5
ADD AX, 4
MOV t4, AX
```

```
;6+7
MOV AX, 6
ADD AX, 7
MOV t5, AX
```

```
;(5+4)*(6+7)
MOV AX, t4
MOV BX, t5
MUL BX
MOV t6, AX
```

```
;2*(3+4)*(8-6)+(5+4)*(6+7)
MOV AX, t3
ADD AX, t6
MOV t7, AX
```

```
;i=2*(3+4)*(8-6)+(5+4)*(6+7)
MOV AX, t7
MOV i_1, AX
```

Mark only one oval.

☐ 1

☐ 2

☒ 3

☐ 4

36. Consider the code code snippet extracted from an implementation of ICG. 2 points
Which of the grammar productions the code is most likely to be associated with?

```
...  
$$->code+=$3->code;  
string l1=newlabel();  
string l2=newlabel();  
$$->code+="cmp "+$3->getname()+" , 0\n";  
$$->code+="je "+l1+"\n";  
$$->code+=$5->code;  
$$->code+="jmp "+l2+"\n";  
$$->code+=l1+":\n";  
$$->code+=$7->code;  
$$->code+=l2+":\n";  
...
```

Mark only one oval.

- ☒ statement : IF LPAREN expression RPAREN statement ELSE statement
- ☐ statement : FOR LPAREN expression_statement expression_statement expression RPAREN statement
- ☐ statement : IF LPAREN expression RPAREN statement
- ☐ statement : WHILE LPAREN expression RPAREN statement

37. Consider the (unoptimized) assembly code snippet below. Which is the smallest sized code that is semantically equivalent to the given code? Assume the temporary variables t0 and t1 and never used outside the given code. 2 points

```
MOV AX, m_1
ADD AX, 0
MOV t0, AX
MOV AX, t0
MOV i_1, AX
MOV AX, n_1
MOV BX, 1
MUL BX
MOV t1, AX
MOV AX, t1
MOV j_1, AX
```

A.

```
MOV AX, m_1
ADD AX, 0
MOV i_1, AX
MOV AX, n_1
MOV BX, 1
MUL BX
MOV j_1, AX
```

C.

```
MOV AX, m_1
MOV i_1, AX
MOV AX, n_1
MOV j_1, AX
```

B.

```
MOV AX, m_1
MOV i_1, AX
MOV AX, n_1
MOV BX, 1
MUL BX
MOV j_1, AX
```

D.

```
MOV AX, m_1
ADD AX, 0
MOV t0, AX
MOV i_1, AX
MOV AX, n_1
MOV t1, AX
MOV j_1, AX
```

Mark only one oval.

☐ A

☐ B

☒ C

☐ D

Google Forms