Test-1

Total points 22/30

Please fill your ID and name correctly in the space provided.

There are thirty questions in this test, each carrying one mark and the duration of the test is 30 minutes. Only one of the many options provided is the correct answer in the given questions. Select the correct answer and move to the next question. You are expected to maintain professional honesty while you attempt the answers to the questions.

Do not forget to submit your responses.

The respondent's email address (**f20181119@pilani.bits-pilani.ac.in**) was recorded on submission of this form.

0 of 0 points

Name *

Shreyas Bhat Kera

ID *

2018A7PS1119P

Questions 1-30 22 of 30 points

✓ We have motivation for studying Principles of programming languages because (1) We will have an enhanced ability to express our ideas (2) We will have a better ability to learn new languages (3) We will be able to create a new programming paradigm. Which of the following statements are true? 1/1

- ○ all statements are true
- [✓ ● Statement 1 and Statement 2 are true] ✓
- ○ None of these
- ○ only Statement 1 is true
- ○ Statement 2 and Statement 3 are true

✗ We have studied about lexemes and tokens. They have key technical difference(s). Which of the following is true with respect to lexemes and tokens? (1.) The lexemes are indeed tokens. In some implementations, they are called lexemes; and tokens in other implementations. (2.) The categorization of lexemes is called tokens (3.) The lexeme of a language is the categorization of tokens. 0/1

- ○ all statements are true
- [✓ ● Statement 2 and Statement 3 are true] ✗
- ○ Statement 1 and Statement 2 are true
- ○ None of these
- ○ only Statement 2 is true

Correct answer

- [✓ ● only Statement 2 is true]

✓ Backus-Naur Form (BNF) is the representation of grammars of a language. Extended BNF (EBNF) is also used for the same purpose. What is the limitation of the Extended BNF notation? (1.) The EBNF is more concise than BNF (2.) We cannot represent the associativity and precedence in EBNF. (3.) It is hard to read EBNF 1/1

- ○ Statement 1 and Statement 2 are true
- ○ None of these
- ○ Statement 2 and Statement 3 are true
- ○ all statements are true
- [✓ only Statement 2 is true] ✓

✓ What is the bottleneck in pure interpretation? 1/1

- [✓ Statement decoding] ✓
- ○ Portability
- ○ Connection between processor and memory
- ○ Movement of instruction from memory to the processor
- ○ None of these

✗ We know that a language has to be readable and an important characteristic of readability is orthogonality. Which of the following statement is correct with respect to orthogonality? 0/1

- [✓ All of the above] ✗
- A relatively small set of primitive constructs can be combined in relatively small number of ways to build the control and data structure of the language.
- The language must provide completely disjoint primitive constructs. These primitive constructs must be least related to each other
- The defined features of the languages must be highly dependent on the context of their appearance in the program.
- None of these

Correct answer

- [✓ A relatively small set of primitive constructs can be combined in relatively small number of ways to build the control and data structure of the language.]

✓ The dangling-else problem occurs 1/1

- when one if-then-else statement is used within another if-then-else statement using curly brackets around itself
- [✓ when one if-then-else statement is used within another if-then-else statement without using curly brackets around itself] ✓
- when a if-then-else statement is used in a program
- none of these

✓ The statements which can change the state of the program are 1/1

- ○ assignment statement, function call statement which returns a value, and for loop statement
- ○ function call statement which does not return a value, return statement and break statement
- [✓ ● function call statement which returns a value, input statement and assignment statement] ✓
- ○ assignment statement, while statement and input statement
- ○ function call statement which does not return a value, input statement and assignment statement
- ○ none of these

✓ The arrays and record structures in C programming language are as described below. Which statement depicts the correct information? 1/1

- [✓ ● Arrays contain elements of same type and allocated contiguous memory while records can contain elements of different types and allocated contiguous memory as well] ✓
- ○ Arrays contain elements of same type and allocated contiguous memory while records can contain elements of different types and allocated non-contiguous memory as well
- ○ Arrays contain elements of different type and allocated contiguous memory while records contain elements of same types and allocated contiguous memory as well
- ○ Arrays contain elements of same type and allocated non-contiguous memory while records can contain elements of different types and allocated contiguous memory
- ○ None of these

✓ A language is modular and has one driver module compulsorily. The program can have any number of modules in it but modules can be positioned only before the driver module. The program may or may not have any module in it. The grammar to define the 'modules' construct of this C-like programming language can be given as below. [The word 'and' in between is to show the two different set of rules in each option] 1/1

- ○ <program>--> <modules> <driverModule> and <modules>--> <oneModule> <modules> | <driverModule> | epsilon
- ○ None of these
- ○ <program>--> <modules> <driverModule> and <modules>--> <oneModule> <modules> | <modules> <oneModule>
- [✓ <program>--> <modules> <driverModule> and <modules>--> <oneModule> <modules> | epsilon] ✓

✓ Parse tree depicts the 1/1

- ○ Grammar rules
- ○ Execution control flow of the program
- [✓ Syntactic structure of the program] ✓
- ○ None of these

✗ The languages which support the complex data type to represent real and imaginary values as a primitive data type are 0/1

- ○ C, Python and Pascal
- ○ Python and C
- ○ None of these
- ○ Python, C and FORTRAN
- ○ FORTRAN and Python
- ○ C and Pascal
- [✓ FORTRAN, Pascal and Python] ✗
- ○ FORTRAN, C and Pascal
- ○ Pascal and FORTRAN

Correct answer

- [✓ FORTRAN and Python]

✗ The APL language is highly writable still it is not used extensively in professional application development. Which of the statement(s) is/are true to support the above statement? (1) The language is having poor documentation which makes it hard to learn. (2) The language is too concise which makes the APL programs difficult to read. (3) The features provided are not sufficient for any application development. 0/1

- ○ Statement 2 and Statement 3 are true
- ● Statement 1 and Statement 2 are true ✗
- ○ None of these
- ○ All statements are true
- ○ only Statement 2 is true

Correct answer

- ● only Statement 2 is true

✓ The grammar that can derive if-then-else statements in a C-like program, which has an essential then-part and optional else-part, is defined using rule <if_statement>--> TK_IF <Condition> TK_THEN { <multiple_statements> } <A> . The rule for the non-terminal <A> to generate optional else-part is given by 1/1

- ○ Both of these
- ○ <A> --> TK_ELSE <multiple_statements> <Condition>
- ○ <A> --> TK_ELSE <Condition> <multiple_statements>
- ● None of these ✓

✓ One of the important aspects of understanding the programming languages is to understand the significance of its implementation. Which of the following statement(s) is/are true with respect to this aspect? (1) We would be able to intelligently use the constructs the way they are designed to be used. (2) We will be in a better position to choose an appropriate construct for the given problem in hand (3) We can debug correctly and write more efficient programs in the programming language. 1/1

- [✓ all statements are true] ✓
- None of these
- Statement 2 and Statement 3 are true
- only Statement 1 is true
- Statement 1 and Statement 2 true

✓ The statements typedef enum { morning, evening, day, night, noon} times; times a,b,c; written in C-language implement the variables a,b and c using 1/1

- 4 memory words
- 15 memory words
- none of these
- 3 memory words
- [✓ 3 bits] ✓
- 15 bits

✓ The dangling else ambiguity can be resolved by (a) using an end marker at the end of each rule for <if_stmt> (b) using the curly bracket pairs around the construct if-then as well as if-then-else. (c) by implementing the association of else clause with the closest if-then clause using static semantics. Which of the following best describe the solution? 1/1

- ○ None of these
- ○ statements (b) and (c)
- ○ statements (a) and (b)
- [✓ All three statements] ✓

✗ Which of the following statement(s) is/are true with respect to the writability of the language? (1) The language that is highly readable is most likely writable (2) The language must provide proper exception handling for enhanced writability (3) The closeness of the language to the desired application also affects the writability. 0/1

- ○ None of these
- ○ Statement 1 and Statement 3 are true
- ○ Statement 1 and Statement 2 are true
- [✓ all statements are true] ✗
- ○ only Statement 1 is true

Correct answer

- [✓ Statement 1 and Statement 3 are true]

✓ The difference between a if-then-else and a switch statement with respect to the flow of execution is that 1/1

- ○ both statements allow only one code segment to be executed at run time.
- [✓ ● The if-then-else statement allows only one of the two code segments executed while the switch statement allows any number of code segments to be executed at run time.] ✓
- ○ The if-then-else statement allows both the two code segments under then and else executed while the switch statement allows only one of the code segments to be executed at run time.
- ○ none of these

✓ The difference between the for-loop constructs of 'C' and ADA programming languages is that 1/1

- ○ The C-for-loop is counter controlled while ADA-for-loop is logic controlled and the loop variable cannot be changed in ADA-for-loop within loop body.
- ○ The C-for-loop is controlled by logic while ADA-for-loop is counter controlled and the loop variable cannot be changed in C-for-loop within loop body.
- [✓ ● The C-for-loop is controlled by logic while ADA-for-loop is counter controlled and the loop variable cannot be changed in ADA-for-loop within loop body.] ✓
- ○ None of these

✗ Once we know the constructs we can simulate/emulate the construct in the language we know. Why do we have to learn different languages to use the constructs supported by these languages? (1) The native language has an elegant and safer way of implementation of the construct. (2) The applied construct will be more cumbersome in other languages as compared to the native language that directly supports that construct. (3) The usage of the construct in other languages would be forced and might not be the most efficient. Which of the following statements are true? 0/1

- ○ only Statement 1 is true
- ○ Statement 1 and Statement 2 true
- ○ all statements are true
- ○ None of these
- [✓ Statement 2 and Statement 3 are true] ✗

Correct answer

- [✓ all statements are true]

✓ The grammar that can derive multiple statements in a program is defined using rule <multiple_statements> --> <single_statement> <multiple statements> which can recursively derive statements. The rule that terminates the recursive generation of the statements while parsing a input, can be best given by 1/1

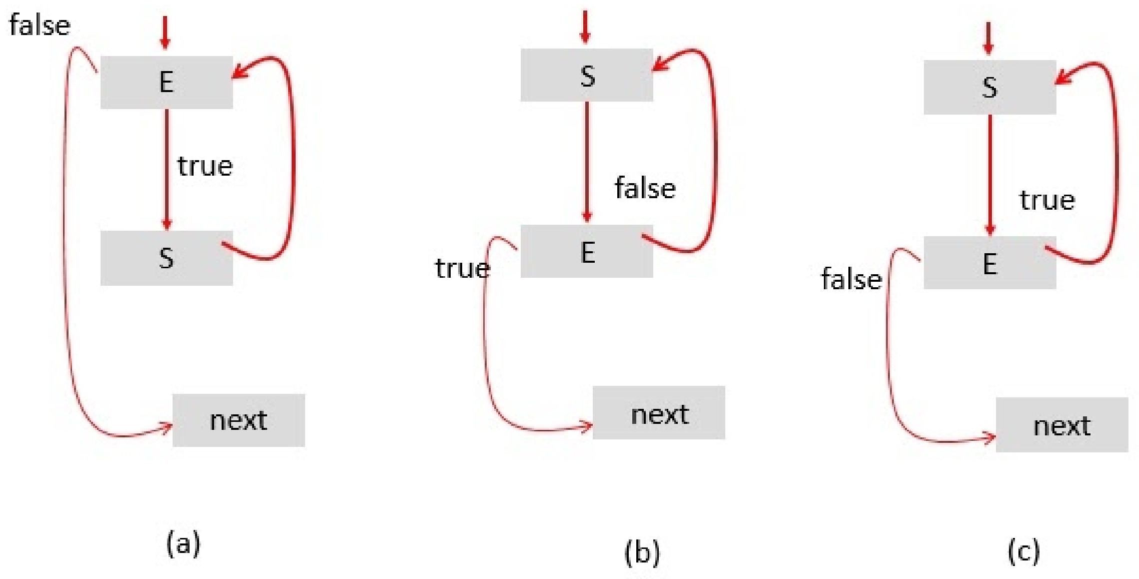
○ None of these

[✓ ● Both of these] ✓

○ <multiple_statements> --> epsilon

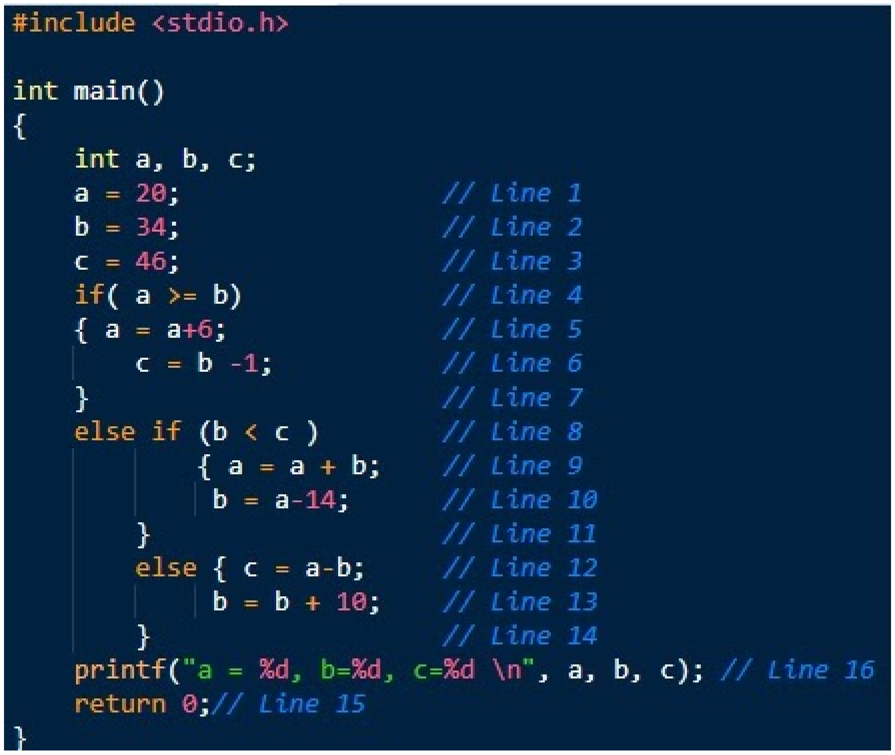
○ <multiple_statements> --> <single_statement>

✓ The flow of execution of the constructs shown in the following figure for (a) (b) and (c) depicts 1/1



- ○ repeat-until, while-do and do-while respectively
- [✓ while-do, repeat-until and do-while respectively] ✓
- ○ while-do, do-while and repeat-until respectively
- ○ do-while, repeat-until and while-do respectively
- ○ repeat-until, do-while and while-do respectively
- ○ none of these
- ○ do-while, while-do and repeat-until respectively

✓ Consider the following code written in C programming language. What is the output of the program? 1/1



```
#include <stdio.h>

int main()
{
    int a, b, c;
    a = 20;                 // Line 1
    b = 34;                 // Line 2
    c = 46;                 // Line 3
    if( a >= b)             // Line 4
    { a = a+6;              // Line 5
        c = b -1;           // Line 6
    }                       // Line 7
    else if (b < c )        // Line 8
            { a = a + b;    // Line 9
             b = a-14;      // Line 10
        }                   // Line 11
        else { c = a-b;     // Line 12
             b = b + 10;    // Line 13
        }                   // Line 14
    printf("a = %d, b=%d, c=%d \n", a, b, c); // Line 16
    return 0;// Line 15
}
```

- none of these
- [✓ a = 54, b=40, c=46] ✓
- a = 26, b=34, c=33
- a = 20, b=44, c=-14

✗ Which of the following language(s) allow the array index type to be a subrange or an enumeration? 0/1

- ○ C
- [✓ ● ADA] ✗
- ○ C and ADA
- ○ C, Pascal and ADA
- ○ Pascal and C
- ○ ADA and Pascal
- ○ None of these
- ○ Pascal

Correct answer

- ● ADA and Pascal

✓ In the compilation process, what is the output of the syntax analyzer? 1/1

- ○ None of these
- ○ Lexical units
- ○ Intermediate code
- [✓ ● Parse trees] ✓
- ○ Machine code

✗ We know that a language has to be readable and an important characteristic of readability is simplicity. Which of the following statement(s) is/are correct with respect to simplicity? (1) Language should not have a large number of basic constructs (2) Language must have the least number of basic constructs without any complex control statements (3) In technical terms, the language must be as close to human spoken language to be categorized as simple. 0/1

- ○ Statement 1 and Statement 2 true
- ○ None of these
- ○ only Statement 1 is true
- ○ Statement 2 and Statement 3 are true
- [✓ all statements are true] ✗

Correct answer

- [✓ only Statement 1 is true]

✓ The grammar to define a single dimensional array element used in any C-program is best given as 1/1

- ○ <element>--> ID SQOP <index> SQCL and <index>-->ID | NUM
- ○ <element>--> ID SQOP <index> SQCL and <index>-->ID
- [✓ <element>--> ID SQOP <index> SQCL and <index>-->ID | NUM | <arithmeticExpn>] ✓
- ○ None of these
- ○ <element>--> ID SQOP ID <index> SQCL and <index>-->NUM

✓ The repeat-until loop construct defined as 'repeat L until C' allows statements in the loop body L to be executed repeatedly until the condition C becomes true. The meaning can be best understood as 1/1

- [✓ L; while not C do L] ✓
- L; while C do L
- while not C do L;
- None of these

✓ The relative offset of an array element A[10], where A is declared as int A[25]; in C-like language is 1/1

- 15*sizeof(int)
- 25*sizeof(int)
- none of these
- [✓ 10*sizeof(int)] ✓

✓ Assembly language is usually not preferred for writing Machine learning applications. Which of the following statement(s) is/are true? (1.) The assembly language is less readable and writable because it has large primitive constructs that can be combined in a large number of ways. (2.) The assembly language is less readable and writable because of the lack of complex constructs in the language. (3.) The assembly language is not designed for machine learning applications. Emulating the constructs of machine learning applications in assembly language makes it less writable and readable. 1/1

- ○ None of these
- ○ only Statement 3 is true
- ● Statement 2 and Statement 3 are true ✓
- ○ all statements are true
- ○ Statement 1 and Statement 2 are true