1. Interpreted programs ____ compiled programs:

a) are not as reliable as.
b) are easily implemented, but they require more space.
c) are faster than.
d) do not allow an easy implementation of many source-level debugging operations.

2. A language is ____ if it performs to its specifications under all conditions:

a) orthogonal
b) readable
c) reliable
d) efficient

3. ____ was created in 1985 by Bjarne Stroustrup:

a) C
b) C++
c) Algol
d) Java

4. Java was developed by ____:

a) Sun in the early 1990s
b) DoD in mid 80s
c) FSF in early 2000
d) MIT in 1980s

5. BNF/EBNF is used for:

a) describing the syntax of formal languages.
b) describing the syntax of programming languages.
c) describing ALGOL programming language.
d) describing the semantics of programming languages.

6. *(6 marks)* Consider the following sequence of a program written in an unknown programming language:
css
Copy code

```
var i, j, k, l, glass, bar, bottle: short;
spirit, liquor, wine, beer, other: double;
```

Construct a **context-free grammar** (in **BNF/EBNF**) such that the above sequence of the program can be generated as a variable declaration.

program ::= { declaration }

declaration ::= "var" variable-list ":" type ";"

variable-list ::= identifier { "," identifier }

identifier ::= letter { letter | digit } letter ::= "a" | "b" | ... | "z" | "A" | "B" | ... | "Z"

digit ::= "0" | "1" | ... | "9"

type ::= "short" | "double"

7. In extended BNF ____ separate alternative parts

a) two vertical bars `||`
b) ampersand `&`
c) vertical bar `|`
d) exclamation sign `!`

---

8. The following Prolog code is computing:

```
he([],[]).
he([H|T],[H]).
```
a) the list containing the first element of a list if the list is non-empty and the empty list, otherwise.
b) the reverse of a list.
c) the tail of a non-empty list.
d) the first element of a list, if the list is non-empty and the empty list, otherwise.

---

9. *(6 marks)* Describe what this Prolog program is computing. Translate into a LISP function.
prolog
Copy code
```
lz([],x,[]).
lz([H|T],H,T).
lz([H|T],A,Y) :- lz(T,A,Y).
```

---

*10. In C, the lifetime of a local variable declared `static` is:

a) the duration of the execution of the whole program.
b) the time from the beginning of the program to the end of the execution of the function where it is declared.
c) the duration of the execution of the function where it is declared.
d) the time from the start of the execution of the function where it is declared to the end of the program.

11. To access global variables in C++:

a) we can use the resolution operator.
b) we can only use pointers.
c) they must have different names than local variables.
d) it is impossible.

---

12. A dynamic binding occurs ____:

a) before run time
b) when the program is loaded into memory
c) during compilation
<u>d)</u> during execution

---

13. For most programming languages, pointers are allocated:

<u>a)</u> on the heap
b) in the main memory
c) before run-time
d) on stack

---

14. A programming language is _____ if type errors are always detected:

<u>a)</u> strongly typed
b) compatible
c) lossy typed
d) easy compiled

---

15. Connectors are allowed for identifiers in COBOL:

<u>a)</u> True
b) False

---

16. Regardless of the programming language, if you know the name of a variable, then:

<u>a)</u> you can use this name to access the variable.
b) you can use this name to determine the type of the variable.
c) you can use this name to assign its value to other variables.
d) you can use this name to assign values to the variable it represents.

---

17. *(6 marks)* Translate the predicate in Prolog into a function in Lisp:
prolog
```
le([],[]).
le([X],X).
le([H|T],Y):- le(T,Y).
```

Ans
```
(defun le (lst)
 (cond
  ((null lst) nil)
   ((null (cdr lst)) (car lst))
  (t (le (cdr lst)))))
```

18. For Lisp, the last value of the following code is:
lisp
Copy code
```
(setq a 5)
(let* ((a 1) (b 2) (c a)) (* (+ a b) c))
```

a) 6
b) 18
c) 1
d) 3

---

19. The operator for exponentiation in Common Lisp is:

a) `**`
b) `~`
c) `expt`
d) does not exist

---

20. *(6 marks)* Translate from Lisp to SCHEME:
lisp
```
(defun f (n)
  (cond
    ((= n 0) 0)
    ((= n 1) 1)
    (t (+ (f (- n 1)) (f (- n 2)))))
)
```

Ans
```
(define (f n)
 (cond
   ((= n 0) 0)
   ((= n 1) 1)
   (else  (+ (f (- n 1)) (f (- n 2))))))
```

---

21. In Scheme, CDR of an empty list is:

a) NIL
b) `()`
c) 0
d) not defined

22. Positive integers k are usually stored in memory as a sequence of n bits representing:

a) k written in base 2 padded on the right with 1 up to n bits, and k<2^n
b) k written in base 2 padded on the right with 0 up to n bits, and 2^k < n
c) k written in base 2 padded on the left with 0 up to n bits, and k<2^n

d) k written in base 2 padded on the left with 1 up to n bits, and k<n

---

23. Stacks can:

a) add elements to the bottom of the stack.
b) cannot be implemented using arrays.
c) be implemented using arrays or linked lists.
d) can only be implemented using linked lists.

---

24. *(10 marks)* What is the main difference between a pointer and an array in C? Examples, at least two.

---

25. Consider the following Pascal sequence:
pascal
Copy code
```
 type
 type1 = array[10..90] of integer;
 type2 = array[9..89] of integer;
 var
   b: type2;
   a: type1;
   i: integer;
 begin
   i := 11;
  a[i] := 5;
  b[i] := 3;
  writeln(b[i])
end.
```

a) The sequence is incorrect because we cannot assign arrays in Pascal.
b) The sequence is correct because `type1` and `type2` are basically the same.
c) The sequence is syntactically correct but will generate a run-time error.
d) The sequence is incorrect because `a` and `b` have different types.

---

26. Consider the following PHP program:
php
Copy code
```
<?php
$i = 10.;
$i++;
```

```
$j = “a$i\n”;
echo "j=$j";
$k = “jojo”;
$k=$i;
echo "k=$k\n";
?>
```

a) The sequence is incorrect because we cannot change the type of `$k` in PHP.
b) The sequence is correct and will print. J = a11 k = 11
c) The sequence is syntactically correct but will generate a run-time error and will not print anything.
d) The sequence is correct and will print. J = a11. k =11.

---

27. Consider the sequence in C:

```
int i;
int *ptri = &i;
free(ptri);
```

a) The sequence is correct because `ptr1` has a non-null value.
b) The sequence is syntactically correct but will generate a run-time error attempting to free a fixed location.
c) The sequence is syntactically incorrect because we cannot free a fixed pointer.
d) The sequence is incorrect because we cannot assign an address of a variable to a pointer.

---

28. Let us assume that we have the following code:
php
Copy code
```
<?php $i=$j?>
```

The value of `$i` is:
a) `""`
b) `null`
c) `0`
d) `undefined`

---

29. *(6 marks)* Give two examples of programming languages where the operator `+` is not commutative. Explain why.

Python

- **Reason**: In Python, the `+` operator is used for **string concatenation** and **list concatenation**, which are order-sensitive.

Java

- **Reason**: In Java, the `+` operator is overloaded to perform **string concatenation** in addition to numeric addition.

30. Primitive data types are:

a) Numerical types, booleans, and character types
b) Integers, records, and pointers
c) Numerical types, decimals, structures, and character types
d) Integers, floating points, objects, and character types

---

31. In Pascal, to initialize a variable with the truth value `true`, we use:

a) `1`
b) `true`
c) `not null`
d) `!nil`

32. C strings are also called:

a) rings
b) arrays
c) zero-terminated strings
d) self-delimiting strings

---

33. In C++, enumeration type can start with:

a) any integer
b) only positive integer
c) only with 0
d) any number

---

34. In C++, indexes in between square brackets can be:

a) only positive integers
b) integers
c) any number
d) of any type

---

35. In Fortran, array indexes are between:

a) forward slashes
b) round parenthesis

c) braces
d) brackets

---

36. In C unions, its variables:

a) store the same value during the entire execution.
b) are allowed to change their types.
c) store the same type value at different times during execution.
d) are allowed to store different type values at different times during execution.

---

37. ____ have built-in set type:

a) C, Pascal, and COBOL
b) Java, ADA, and Pascal
c) Java, Modula, and Pascal
d) Ada, Modula, and Fortran 77

---

38. For a C program, all variables are initialized with `0`:

a) True
b) False

---

39. A C program consists of:

a) methods and variables and one function must be the `main` method.
b) declaration of functions and variables, and one function must be the `main` function.
c) declaration of methods and variables.
d) declaration of functions and one function must be the `Main` function.

---

40. Directives begin in C with:

a) `%`
b) `//`
c) `#`
d) `define`

---

41. For C functions, parameters are passed:

a) by address
b) by value

c) by reference
d) by name

---

42. C and C++:

a) are both case sensitive.
b) are not case sensitive for directives but case sensitive for everything else.
c) are case sensitive in UNIX implementations and not case sensitive in Microsoft Windows implementations.

---

43. In C++ functions having the same name:

a) must have a different number of parameters, but common parameters must be of the same type.
b) must have the same number of parameters.
c) must have either a number of parameters or different type parameters.
d) must have the same return type.

44. In C++, last parameters:

a) must have implicit values.
b) must be integers.
c) are initialized with 0 by default.
d) can have implicit values.

---

45. In C++, to access a variable in a specific namespace:

a) you must use the namespace name, the pointer dereference (`->`) operator, and the name of the variable.
b) you must use the namespace name, the star (`*`) operator, and the name of the variable.
c) you must use the namespace name, the dot (`.`) operator, and the name of the variable.
d) you must use the namespace name, the resolution (`::`) operator, and the name of the variable.

---

46. In C++, `cout` is:

a) the name of the standard input.
b) an object of the `iostream` class in the `std` namespace.
c) the name of the standard output.
d) a global object in the `iostream` class.

---

47. In C++, the `&` sign after a type is used to declare:

a) a global variable
b) an address

c) a reference
d) a pointer

---

48. If an operand `#` is right associative, the expression `a # b # c # d` is equivalent to:

a) `((a # b) # c) # d`
b) `a # (b # (c # d))`
c) `(a # b) # (c # d)`
d) None of the others

---

49. _____ cannot be overloaded in C++:

a) `()` operator
b) `[]` operator
c) ternary operator
d) `,` operator

---

50. Typically, _____ operators have a higher precedence than binary arithmetic operators:

a) sequential
b) unary
c) ternary
d) binary relational

---

51. _____ have the highest precedence:

a) `.` operators
b) Parentheses
c) Unary operators
d) Sequential operators

52. For C++, by overloading an operator, we cannot:

a) change the type of the first operand unless we change the associativity order.
b) change the priority level.
c) change the type of the second operand if we use a friend function.
d) change the type of both operands.

---

53. For C++, by overloading an operator, we cannot:

a) change the type of the first operand if we use a friend function.
b) change the arity.

c) change the type of the second operand if we use a member function.
d) change the type of operands.

---

54. A functional side effect occurs when:

a) a function returns a pointer.
b) a function changes a pass-by-value parameter.
c) a function changes a local variable.
d) a function changes a two-way parameter or a nonlocal variable.

---

55. To avoid functional side effects, you can:

a) disallow two-way parameters in functions.
b) disallow nonlocal references in functions.
c) demand that operand evaluation order to be fixed.
d) all of the above.

---

56. When overloading operators, you should:

a) consider short-circuit evaluation.
b) ignore the priority of operators.
c) avoid creating ambiguity.
d) only overload arithmetic operators.

---

57. Assume `a=3`, `b=6`, `c=5`, and `d=2` are four integer variables in C. The value of `f` after the expression `f=a < b, c < d` is:

a) 1
b) false
c) 3
d) 0

---

58. ____ is a compound statement that can define a new scope:

a) pretest statement
b) label
c) control structure
d) block

---

59. Pascal's rule for `if`:

a) `else` goes with the first `then`.
b) a semicolon is required before `else`.
c) `endif` closes the `if` without `else`.
d) `else` goes with the nearest `then`.

---

60. The first multiple selection construct was:

a) ALGOL's case statement
b) FORTRAN arithmetic IF
c) C's switch statement
d) COBOL Select statement

---

61. For statement `switch` in C:

a) the next case is executed if `break` is present.
b) the default statement is executed after the case block is executed.
c) the next case is executed if `break` is not present.
d) the next statement after `switch` is executed after the case block is executed.

---

*62. For Pascal, loop variables for counter control loops:

a) can be int or real.
b) must be of an ordinal type of usual scope.
c) can be of any type.
d) can be changed in the loop.

---

*63. For C loop variables for a `for` statement:

a) the loop parameters can be changed, but they are evaluated just once.
b) loop var cannot be changed in the loop.
c) the loop variables must be of an ordinal type of usual scope.
d) everything can be changed in the loop.

---

64. `do-while` and `repeat-until` are usually ____ statements:

a) multiple selection
b) counting loop
c) post-test logical loop
d) pre-test logical loop

*65. Functions provide:

a) user-defined statements
b) user-defined operators
c) a constant value
d) user-defined variables

---

66. A(n) ____ is a dummy variable listed in the subprogram header and used in the subprogram:

a) parameter profile
b) actual parameter
c) formal parameter
d) declaration

---

67. A(n) ____ represents a value or address used in the subprogram call statement:

a) actual parameter
b) formal parameter
c) declaration
d) parameter profile

---

*68. A(n) ____ of a subprogram is the number, order, and types of its parameters:

a) parameter profile
b) formal parameter
c) declaration
d) subprogram declaration

---

69. If local variables are ____, subprograms cannot be history sensitive:

a) heap-dynamic
b) global
c) stack-dynamic
d) static

---

70. In C++, the transfer of parameters is done by:

a) pass-by-reference only
b) pass-by-name
c) pass-by-value or pass-by-reference
d) pass-by-value-result

---

71. In LISP, the transfer of parameters is done by:

a) pass-by-name
b) pass-by-value-result
c) pass-by-reference only
d) pass-by-value or pass-by-reference

---

72. Type checking of parameters is always required in:

a) Pascal, Java, and Ada
b) FORTRAN 77 and original C
c) ANSI C and C++
d) PHP and Java

---

73. A(n) ____ subprogram is one that takes parameters of different types on different activations:

a) functional
b) virtual
c) polymorphic
d) overloaded

---

74. C allows any return type except:

a) functions and arrays
b) functions and pointers
c) structures and functions
d) pointers and arrays

---

75. The ____ variables of a subprogram are those that are visible but not declared in the subprogram:

a) local
b) global
c) nonlocal
d) static

---

76. In C++, objects are copied ____ by default:

a) calling the constructors
b) using the base class constructors
c) calling the implicit constructor
d) bit by bit

---

77. In C++, if objects have as members other objects or pointers to copy them, it is recommended to:

a) define the copy constructor
b) overload the implicit constructor
c) call the implicit constructor
d) call a constructor first

---

78. In C++, if we do not declare any constructor:

a) compiling will fail.
b) the implicit constructor is automatically created by the compiler.
c) no constructor is available.
d) compiling will succeed, but we may get a run-time error.

---

79. In C++ `struct`:

a) is similar to `class`, but `struct` has no functions as members.
b) is similar to `class`, but `struct` has public members.
c) cannot be used to define a class.
d) is similar to `class`, but `struct` has private members.

---

80. *(8 marks)* List four languages presented in class by students and explain what you liked and what you did not like for each of them.