MCQS AND ONE LINE QUESTIONS

C PROGRAMMING

Disclaimer:

The questions provided in this material are meant solely for practice and learning purposes. They are designed to help students revise concepts, test their understanding, and improve problem-solving skills. There is no guarantee that these questions will appear in any exam or that they will be in the same format. This material is intended to guide your preparation and encourage comprehensive study of the subject. Students are advised to cover the entire syllabus thoroughly and use this material as a tool to strengthen their knowledge and readiness.

MCQS

UNIT 1: Introduction to Computers and Programming Concepts

Q1. Which of the following is the correct sequence of data processing in a computer?
a) Input → Storage → Output → Processing
b) Input → Processing → Output → Storage
c) Input → Processing → Storage → Output
d) Input → Processing → Output

Answer: d) Input → Processing → Output

Q2. Which component of the computer is responsible for performing arithmetic and logical operations?
a) Control Unit
b) ALU
c) Memory Unit
d) Input Device

Answer: b) ALU

Q3. Hardware and software are related because:
a) Hardware is useless without software
b) Software is useless without hardware
c) Both are dependent on each other
d) All of the above

Answer: d) All of the above

Q4. Which of the following is **not** a system software?
a) Operating System
b) Compiler
c) Antivirus
d) MS Word

Answer: d) MS Word

Q5. A compiler translates:
a) One instruction at a time into machine code
b) The entire program into machine code at once
c) High-level language directly into source code
d) Machine code into high-level language

Answer: b) The entire program into machine code at once

Q6. Which of the following is **faster in execution**?
a) Machine-level language
b) Assembly language
c) High-level language
d) None of the above

Answer: a) Machine-level language

Q7. Flowcharts are mainly used for:
a) Writing source code in a structured way
b) Representing logical steps of a program pictorially
c) Debugging syntax errors in programs
d) Storing data in graphical form

Answer: b) Representing logical steps of a program pictorially

Q8. An algorithm is:
a) A computer program written in C
b) A sequence of well-defined steps to solve a problem
c) A graphical representation of data
d) A debugging tool

Answer: b) A sequence of well-defined steps to solve a problem

Q9. Which language is the closest to machine hardware?
a) Assembly language
b) High-level language
c) Machine language
d) Java

Answer: c) Machine language

Q10. Which of the following best describes application software?
a) Controls overall operations of computer
b) Provides a platform for running programs
c) Helps the user perform specific tasks
d) Translates programs into machine code

Answer: c) Helps the user perform specific tasks

Q11. The control unit of a computer is responsible for:
a) Performing arithmetic operations
b) Storing program instructions
c) Directing the flow of data and instructions
d) Printing the final output

Answer: c) Directing the flow of data and instructions

Q12. Which one is **not** an input device?
a) Keyboard
b) Mouse
c) Scanner
d) Printer

Answer: d) Printer

Q13. The software that manages computer hardware and provides services for application software is:
a) Compiler
b) Operating System
c) Interpreter
d) Assembler

Answer: b) Operating System

Q14. A high-level language program is first converted into:
a) Binary code
b) Flowchart
c) Source code
d) Object code

Answer: d) Object code

Q15. Which of the following best describes an interpreter?
a) Translates an entire program at once
b) Translates one statement at a time
c) Converts binary code into assembly language
d) Works only with operating systems

Answer: b) Translates one statement at a time

Q16. Which of the following is an **example of system software**?
a) MS Excel
b) Browser
c) Linux
d) Photoshop

Answer: c) Linux

Q17. Which of the following is true about machine language?
a) It is user-friendly
b) It is portable across different systems
c) It is expressed in binary codes (0s and 1s)
d) It does not require a compiler or interpreter

Answer: c) It is expressed in binary codes (0s and 1s)

Q18. In a flowchart, a **decision** is represented by which symbol?
a) Rectangle
b) Diamond
c) Oval
d) Parallelogram

Answer: b) Diamond

Q19. Which of the following is **not** an example of application software?
a) MS Word
b) Tally ERP
c) C Compiler
d) Photoshop

Answer: c) C Compiler

Q20. The step-by-step procedure to solve a problem that is independent of programming language is called:
a) Program
b) Algorithm
c) Flowchart
d) Source code

Answer: b) Algorithm

Q21. Which of the following performs the job of translating assembly language into machine code?
a) Compiler
b) Interpreter
c) Assembler
d) Loader

Answer: c) Assembler

Q22. Which of the following is an example of utility software?
a) Word Processor
b) Antivirus Program
c) Compiler
d) Interpreter

Answer: b) Antivirus Program

Q23. The "brain" of the computer is considered to be:
a) ALU
b) CPU
c) Control Unit
d) RAM

Answer: b) CPU

Q24. Which type of memory is volatile?
a) Hard Disk
b) RAM
c) ROM
d) Flash Memory

Answer: b) RAM

Q25. In flowcharts, an **oval shape** is used to represent:
a) Decision
b) Process
c) Start/End
d) Input/Output

Answer: c) Start/End

Q26. The difference between low-level and high-level languages is:
a) Low-level is easier to understand than high-level
b) High-level languages are machine dependent
c) Low-level languages are closer to hardware
d) High-level languages are written in binary

Answer: c) Low-level languages are closer to hardware

Q27. Which of the following is a disadvantage of using machine-level language?
a) Faster execution
b) Easy to learn
c) Difficult to debug and maintain
d) No need of translators

Answer: c) Difficult to debug and maintain

Q28. Which of the following is an **example of an application software package**?
a) Operating System
b) MS Office
c) Compiler
d) Device Driver

Answer: b) MS Office

Q29. An algorithm must have:
a) Finite number of steps
b) Only mathematical calculations
c) Flowchart representation
d) Compiler for execution

Answer: a) Finite number of steps

Q30. Which of the following is true about a compiler?
a) It executes the program line by line
b) It translates high-level language into assembly code
c) It requires the whole program before translation
d) It produces faster execution than an interpreter

Answer: c) It requires the whole program before translation

Q31. Which type of software helps the user perform specific tasks such as document writing or data analysis?
a) System Software
b) Application Software
c) Utility Software
d) Programming Software

Answer: b) Application Software

Q32. Which of the following is **not a programming language type**?
a) High-level language
b) Assembly-level language
c) Machine-level language
d) Operating language

Answer: d) Operating language

Q33. Which of the following is true for system software?
a) It directly assists users in performing tasks
b) It manages and controls computer hardware
c) It is optional for computer operation
d) It is always written in high-level language

Answer: b) It manages and controls computer hardware

Q34. The process of designing an algorithm before writing a program helps to:
a) Increase syntax errors
b) Reduce program efficiency
c) Solve the problem logically and clearly
d) Replace programming language

Answer: c) Solve the problem logically and clearly

Q35. A program that translates high-level language statements into machine code and executes them line by line is called:
a) Compiler
b) Interpreter
c) Assembler
d) Linker

Answer: b) Interpreter

Q36. Which memory type stores data permanently even when the power is turned off?
a) RAM
b) Cache
c) ROM
d) Registers

Answer: c) ROM

Q37. In a flowchart, which symbol represents input or output operation?
a) Rectangle
b) Diamond
c) Parallelogram
d) Oval

Answer: c) Parallelogram

Q38. Which of the following best describes a **high-level language**?
a) Written in binary
b) Machine dependent
c) Close to natural language and easier to understand
d) Directly executed by hardware

Answer: c) Close to natural language and easier to understand

Q39. Which of the following is **not a function of the CPU**?
a) Control instructions execution
b) Perform arithmetic and logic operations
c) Store data permanently
d) Fetch instructions from memory

Answer: c) Store data permanently

Q40. The main difference between a compiler and an interpreter is:
a) Compiler translates entire program before execution, interpreter translates line by line
b) Compiler is slower in execution than interpreter
c) Interpreter produces object code, compiler does not
d) Compiler works only on system software

Answer: a) Compiler translates entire program before execution, interpreter translates line by line

UNIT 2: Introduction to C Language and Program Structure

Q1. Which of the following is **not a feature of C language**?
a) Structured language
b) Low-level language
c) Portable
d) Supports recursion

Answer: b) Low-level language

Q2. The correct structure of a simple C program includes:
a) Header files → Main function → Comments → Statements
b) Comments → Header files → Main function → Statements
c) Header files → Main function → Statements → Return statement
d) Main function → Header files → Statements → Return statement

Answer: c) Header files → Main function → Statements → Return statement

Q3. Which of the following is used to include standard library functions in C?
a) #include
b) #define
c) import
d) using

Answer: a) #include

Q4. What is the purpose of comments in a C program?
a) To execute additional instructions
b) To explain the code for better understanding
c) To optimize code execution
d) To define constants

Answer: b) To explain the code for better understanding

Q5. Which of the following is a valid integer constant in C?
a) 12.5
b) 'a'
c) 100
d) "100"

Answer: c) 100

Q6. Which of the following is a valid declaration of a float variable?
a) float x;
b) int x;
c) char x;
d) double x;

Answer: a) float x;

Q7. Which operator in C is used to find the remainder after division?
a) /
b) %
c) *
d) //

Answer: b) %

Q8. Which of the following expressions evaluates to 1 in C?
a) 5 > 3
b) 5 < 3
c) 5 == 3
d) 5 != 5

Answer: a) 5 > 3

Q9. In C, implicit type conversion is also called:
a) Casting
b) Type promotion
c) Type demotion
d) Parsing

Answer: b) Type promotion

Q10. Which operator has the **highest precedence** among the following?
a) +
b) *
c) ==
d) &&

Answer: b) *

Q11. Which of the following is a valid C comment?
a) /* This is a comment */
b) // This is a comment

c) Both a and b
d) # This is a comment

Answer: c) Both a and b

Q12. Which of the following is **not a valid data type in C**?
a) int
b) float
c) decimal
d) char

Answer: c) decimal

Q13. Which of the following is a **constant** in C?
a) int x = 10;
b) #define PI 3.14
c) float y;
d) char ch;

Answer: b) #define PI 3.14

Q14. The operator used for logical AND in C is:
a) &
b) &&
c) ||
d) !

Answer: b) &&

Q15. Which of the following is **not a valid variable name** in C?
a) myVar
b) _count
c) 1stValue
d) total_sum

Answer: c) 1stValue

Q16. In C, what is the default type of a real constant like 3.5?
a) int
b) float
c) double
d) char

Answer: c) double

Q17. Which of the following statements is true about type casting in C?
a) It changes the value of a variable permanently
b) It temporarily converts one data type to another
c) It is only applicable to integer variables
d) It is done using the operator =

Answer: b) It temporarily converts one data type to another

Q18. Which of the following expressions evaluates to 0 in C?
a) 5 < 3
b) 5 > 3
c) 5 == 5
d) 3 != 3

Answer: a) 5 < 3

Q19. In C, which operator has **higher precedence**?
a) +
b) -
c) *
d) ==

Answer: c) *

Q20. Which of the following is **true about C language**?
a) C is a purely object-oriented language
b) C does not support structured programming
c) C is a general-purpose, structured programming language
d) C can be executed without compilation

Answer: c) C is a general-purpose, structured programming language

Q21. Which of the following is a **relational operator** in C?
a) +
b) !=
c) %
d) &&

Answer: b) !=

Q22. Which of the following is **used to combine two conditions** in C?
a) &
b) &&
c) %
d) !

Answer: b) &&

Q23. The expression `int x = 5 / 2;` will assign what value to `x`?
a) 2.5
b) 2
c) 3
d) 5

Answer: b) 2

Q24. What is the output of `printf("%d", (int)3.7);`?
a) 3
b) 3.7
c) 4
d) 0

Answer: a) 3

Q25. In the expression `a + b * c`, which operation is performed first according to precedence rules?
a) Addition
b) Multiplication
c) Both simultaneously
d) Cannot determine

Answer: b) Multiplication

Q26. Which operator is used for **logical NOT** in C?
a) !
b) ~
c) ^
d) !=

Answer: a) !

Q27. Which of the following will **promote a float to double** automatically?
a) Implicit type conversion
b) Explicit type conversion
c) Casting
d) None of the above

Answer: a) Implicit type conversion

Q28. Which of the following statements is **true about C constants**?
a) Constants can be changed during execution
b) Constants are declared using `#define` or `const` keyword
c) Constants are same as variables
d) Constants are only used for strings

Answer: b) Constants are declared using `#define` or `const` keyword

Q29. Which of the following operators has **lowest precedence**?
a) *
b) +
c) ==
d) &&

Answer: d) &&

Q30. In C, which of the following **type conversions** is explicit?
a) int to float automatically
b) float to int using `(int)` casting
c) char to int automatically
d) short to int automatically

Answer: b) float to int using `(int)` casting

Q31. Which of the following **header files** is required for `printf()` and `scanf()` functions?
a) stdlib.h
b) stdio.h
c) conio.h
d) math.h

Answer: b) stdio.h

Q32. Which of the following is a **valid C variable declaration**?
a) int 2ndNumber;
b) float total_marks;
c) char my-char;
d) int #count;

Answer: b) float total_marks;

Q33. What is the result of the expression `5 + 3 * 2` in C?
a) 16
b) 11
c) 10
d) 8

Answer: b) 11

Q34. Which of the following is **not a valid constant type** in C?
a) Integer constant
b) Floating constant
c) String constant
d) Dynamic constant

Answer: d) Dynamic constant

Q35. Which operator is used to **access the value at a pointer address**?
a) &
b) *
c) ->
d) %

Answer: b) *

Q36. Which of the following statements about **C operators** is correct?
a) Unary operators take two operands
b) Binary operators take one operand
c) Ternary operator takes three operands
d) Assignment operator is ternary

Answer: c) Ternary operator takes three operands

Q37. Which of the following **type conversions** is done automatically by C?
a) int to char
b) float to double
c) double to int
d) char to float

Answer: b) float to double

Q38. The operator `==` in C is used for:
a) Assignment
b) Equality comparison
c) Increment
d) Modulus

Answer: b) Equality comparison

Q39. Which of the following **is evaluated first** in the expression `a + b * c / d`?
a) a + b
b) b * c
c) c / d
d) Left to right

Answer: b) b * c

Q40. Which of the following statements about **C program comments** is true?
a) Comments are executed by the compiler
b) Comments can be single-line or multi-line
c) Comments can only appear at the start of the program
d) Comments are mandatory in every C program

Answer: b) Comments can be single-line or multi-line

UNIT 3: Control Structures in C

Q1. Which of the following is a **decision-making statement** in C?
a) for
b) if
c) while
d) continue

Answer: b) if

Q2. In C, the `if-else` statement is used to:
a) Repeat a block of code
b) Choose between two alternatives
c) Jump to another part of the program
d) None of the above

Answer: b) Choose between two alternatives

Q3. Which of the following is true about **nested if statements**?
a) One if statement inside another if statement
b) Only two if statements are allowed
c) Cannot be used with else
d) Only for loop can be nested

Answer: a) One if statement inside another if statement

Q4. Which of the following statements is **true about switch-case**?
a) Can evaluate string expressions
b) Used for multi-way branching
c) Requires boolean condition
d) Can replace all if-else statements

Answer: b) Used for multi-way branching

Q5. What is the default behavior if `break` is not used in a switch-case?
a) Exit the switch automatically
b) Execute the next case statements until break or end
c) Compiler error
d) Skip all cases

Answer: b) Execute the next case statements until break or end

Q6. Which loop in C **executes at least once**?
a) for
b) while
c) do-while
d) if-else

Answer: c) do-while

Q7. Which of the following loops is used when the **number of iterations is known**?
a) while
b) do-while
c) for
d) goto

Answer: c) for

Q8. Which statement is used to **exit a loop immediately**?
a) continue
b) break
c) goto
d) return

Answer: b) break

Q9. The `continue` statement in a loop is used to:
a) Exit the loop
b) Skip the remaining statements in the current iteration and move to next iteration
c) Stop the program
d) Restart the program

Answer: b) Skip the remaining statements in the current iteration and move to next iteration

Q10. Which of the following statements about `goto` is true?
a) It jumps to a labeled statement in the program
b) It is recommended in structured programming
c) Cannot be used in loops
d) Only works with switch-case

Answer: a) It jumps to a labeled statement in the program

Q11. What will be the output of the following code?

```
int x = 5;
if(x > 0)
    if(x < 10)
        printf("A");
    else
        printf("B");
```

a) A
b) B
c) AB
d) Error

Answer: a) A

Q12. Which of the following is a **multi-way decision-making statement**?
a) if
b) if-else
c) switch-case
d) while

Answer: c) switch-case

Q13. In a `for` loop, the **initialization part** is executed:
a) Once before the loop starts
b) Every iteration
c) After the loop ends
d) Only when condition is false

Answer: a) Once before the loop starts

Q14. In a `while` loop, the condition is checked:
a) Before executing the loop body
b) After executing the loop body
c) Only once
d) Only if break is used

Answer: a) Before executing the loop body

Q15. What will be the output of this code?

```
for(int i=1; i<=3; i++){
    if(i==2)
        continue;
    printf("%d", i);
}
```

a) 123
b) 13
c) 12
d) 23

Answer: b) 13

Q16. Which of the following loops is most suitable when the **condition must be checked after executing the loop body at least once**?
a) for
b) while
c) do-while
d) switch

Answer: c) do-while

Q17. What is the output of the following snippet?

```
int i=1;
while(i<4){
    printf("%d",i);
    i++;
}
```

a) 1 2 3 4
b) 1 2 3
c) 0 1 2
d) Infinite loop

Answer: b) 1 2 3

Q18. Nested loops are used when:
a) A loop is placed inside another loop
b) Loops are executed sequentially
c) Only one iteration is needed
d) Using switch-case

Answer: a) A loop is placed inside another loop

Q19. Which of the following is **true about break statement in nested loops**?
a) Exits only the current loop
b) Exits all loops
c) Skips one iteration
d) Jumps to a labeled statement

Answer: a) Exits only the current loop

Q20. What will be the output of this code?

```
int i=0;

do{

    printf("%d", i);

    i++;

}while(i<0);
```

a) 0
b) Nothing
c) 1
d) Infinite loop

Answer: a) 0

Q21. What will be the output of the following code?

```
int i;
for(i=0; i<3; i++){
    for(int j=0; j<2; j++){
        printf("%d", i+j);
    }
}
```

a) 001122
b) 011223
c) 012123
d) 012345

Answer: c) 012123

Q22. Which statement is **not recommended** in structured programming?
a) break
b) continue
c) goto
d) for

Answer: c) goto

Q23. In switch-case, the `default` case:
a) Must be used
b) Executes only if no other case matches
c) Can only appear at the beginning
d) Can be used multiple times

Answer: b) Executes only if no other case matches

Q24. Which of the following is **true for if-else ladder**?
a) All conditions are checked regardless of previous results
b) Only the first true condition executes

c) Else can appear without if
d) Nested if cannot be used inside it

Answer: b) Only the first true condition executes

Q25. Which of the following **modifies loop execution without exiting the loop**?
a) break
b) continue
c) goto
d) return

Answer: b) continue

Q26. The output of this code is:

```
int x=1;

switch(x){

    case 1: printf("A");

    case 2: printf("B");

    default: printf("C");

}
```

a) A
b) AB
c) ABC
d) AC

Answer: c) ABC

Q27. Which loop is best when **the number of iterations is unknown** and condition is tested at the start?
a) for
b) while
c) do-while
d) switch

Answer: b) while

Q28. What will be the output of the following snippet?

```
for(int i=0;i<3;i++){
    if(i==1) break;
    printf("%d",i);
}
```

a) 0 1
b) 0
c) 1
d) 0 1 2

Answer: b) 0

Q29. Which of the following is **true about nested loops**?
a) Only two loops can be nested
b) Inner loop completes all its iterations for every outer loop iteration
c) Inner loop executes once
d) Nested loops are not allowed in C

Answer: b) Inner loop completes all its iterations for every outer loop iteration

Q30. What is the purpose of `goto` in C?
a) To execute a block of code repeatedly
b) To skip current iteration
c) To jump to a labeled statement anywhere in the function
d) To exit the program

Answer: c) To jump to a labeled statement anywhere in the function

Q31. Which of the following is the correct syntax for a `for` loop in C?
a) for(i=0; i<10; i++) { }
b) for(i=0; i<10; i--) { }
c) for(i<10; i=0; i++) { }
d) for(i=0; i<10) { }

Answer: a) for(i=0; i<10; i++) { }

Q32. What will be the output of:

```
int i=0;
do{
    printf("%d", i);
    i++;
}while(i<3);
```

a) 0 1 2
b) 1 2 3
c) 0 1 2 3
d) Infinite loop

Answer: a) 0 1 2

Q33. Which of the following statements about `break` is correct?
a) It exits the nearest enclosing loop or switch
b) It skips the current iteration of a loop
c) It jumps to a labeled statement
d) It stops the entire program

Answer: a) It exits the nearest enclosing loop or switch

Q34. In a `switch` statement, which of the following is **mandatory**?
a) case statements
b) default statement
c) break statement
d) None of the above

Answer: a) case statements

Q35. What will the following code print?

```
for(int i=0;i<5;i++){
    if(i==3) break;
```

```
    printf("%d",i);
}
```

a) 0 1 2 3 4
b) 0 1 2
c) 1 2 3
d) 0 1 2 3

Answer: b) 0 1 2

Q36. Which of the following loops is used when the **condition must be checked before the loop executes**?
a) do-while
b) while
c) for
d) both b and c

Answer: d) both b and c

Q37. Which of the following is true about nested if statements?
a) Else belongs to the nearest unmatched if
b) Else can belong to any if
c) Nested if cannot have else
d) Nested if is not allowed in C

Answer: a) Else belongs to the nearest unmatched if

Q38. What will be the output of the following code?

```
int x=2;
switch(x){
    case 1: printf("One");
    case 2: printf("Two");
    default: printf("Default");
}
```

a) Two
b) Two Default
c) 2
d) Error

Answer: b) Two Default

Q39. Which of the following statements is **true about `continue`** in nested loops?
a) Exits all loops
b) Skips the remaining statements in the current iteration of the **innermost loop**
c) Terminates the program
d) Jumps to the start of the outer loop

Answer: b) Skips the remaining statements in the current iteration of the **innermost loop**

Q40. The `goto` statement can be used to:
a) Jump to any label within the same function
b) Jump to another function
c) Replace break statement
d) Replace continue statement

Answer: a) Jump to any label within the same function

UNIT 4: Arrays and Strings

Q1. Which of the following best describes an array in C?
a) A collection of variables of different data types
b) A collection of variables of the same data type stored at contiguous memory locations
c) A single variable storing multiple values
d) A pointer to a variable

Answer: b) A collection of variables of the same data type stored at contiguous memory locations

Q2. Which of the following is the correct way to declare a one-dimensional array of 10 integers?
a) int arr[10];
b) int arr;
c) int[10] arr;
d) int arr();

Answer: a) int arr[10];

Q3. Which of the following is the correct initialization of an array in C?
a) int arr[3] = {1,2,3};
b) int arr = {1,2,3};
c) int arr(3) = {1,2,3};
d) int arr[3]; arr = {1,2,3};

Answer: a) int arr[3] = {1,2,3};

Q4. What is the index of the **first element** in an array in C?
a) -1
b) 0
c) 1
d) Depends on compiler

Answer: b) 0

Q5. How many elements are there in the array declared as `int arr[5];`?
a) 4
b) 5
c) 6
d) Depends on initialization

Answer: b) 5

Q6. Which of the following correctly declares a **two-dimensional array** of 3 rows and 4 columns?
a) int arr[3,4];
b) int arr[3][4];
c) int arr(3)(4);
d) int arr[4][3];

Answer: b) int arr[3][4];

Q7. Which of the following is **true about arrays in C**?
a) Size of an array can be changed at runtime
b) Arrays store elements of the same data type
c) Array elements are stored randomly in memory
d) Arrays can hold variables of different types

Answer: b) Arrays store elements of the same data type

Q8. Which of the following is the correct way to declare a string in C?
a) char str[10];
b) string str[10];
c) char str;
d) string str;

Answer: a) char str[10];

Q9. Which built-in function is used to find the **length of a string** in C?
a) strlen()
b) strlength()
c) sizeof()
d) strsize()

Answer: a) strlen()

Q10. What is the output of the following code?

```
char str[6] = "Hello";
printf("%c", str[1]);
```

a) H
b) e
c) l
d) o

Answer: b) e

Q11. Which of the following statements correctly initializes a 2D array?

a) `int arr[2][3] = {{1,2,3},{4,5,6}};`
b) `int arr[2][3] = {1,2,3,4,5,6};`
c) Both a and b
d) None of the above

Answer: c) Both a and b

Q12. What is the value of `arr[1][2]` in the following code?

```
int arr[2][3] = {{1,2,3},{4,5,6}};
```

a) 3
b) 4
c) 5
d) 6

Answer: c) 5

Q13. Which of the following is true about string storage in C?

a) Strings are arrays of characters terminated by `\0`
b) Strings can contain any data type
c) Strings do not need a null terminator
d) Strings cannot be initialized

Answer: a) Strings are arrays of characters terminated by `\0`

Q14. Which function **copies one string to another** in C?

a) strcpy()
b) strcat()
c) strcmp()
d) strlen()

Answer: a) strcpy()

Q15. Which function **concatenates two strings** in C?

a) strcat()
b) strcmp()
c) strcpy()
d) strlen()

Answer: a) strcat()

Q16. Which of the following is true about array indexing in C?
a) Indexing starts from 1
b) Indexing starts from 0
c) Indexing starts from -1
d) Indexing starts from 2

Answer: b) Indexing starts from 0

Q17. What will be the output of the following code?

```
char str[] = "CProgramming";
printf("%c", str[0]);
```

a) C
b) P
c) r
d) G

Answer: a) C

Q18. Which function is used to **compare two strings** in C?
a) strcmp()
b) strcat()
c) strcpy()
d) strlength()

Answer: a) strcmp()

Q19. Which of the following is **correct syntax** for initializing a 1D array?
a) `int arr[5] = {1,2,3,4,5};`
b) `int arr[] = {1,2,3,4,5};`
c) Both a and b
d) `int arr = {1,2,3,4,5};`

Answer: c) Both a and b

Q20. Which of the following is the **correct way to declare a 2D character array for strings**?
a) `char names[3][10];`
b) `char names[10][3];`

c) `char names[3][];`
d) `char names[][10];`

Answer: a) `char names[3][10];`

Q21. Which of the following **built-in functions** converts a string to lowercase in C?
a) strlwr()
b) strupr()
c) strcpy()
d) strcat()

Answer: a) strlwr()

Q22. Which of the following **built-in functions** converts a string to uppercase in C?
a) strupr()
b) strlwr()
c) strcat()
d) strcpy()

Answer: a) strupr()

Q23. Which of the following statements about 2D arrays is **true**?
a) All rows must have the same number of columns
b) Each row can have different number of columns
c) 2D arrays can store different data types
d) 2D arrays cannot be initialized

Answer: a) All rows must have the same number of columns

Q24. What will be the output of the following code?

```
char str[6] = "Hello";
printf("%s", str);
```

a) H
b) Hello
c) e
d) Error

Answer: b) Hello

Q25. Which function **calculates the length of a string excluding the null character**?
a) strlen()
b) sizeof()
c) strsize()
d) strcmp()

Answer: a) strlen()

Q26. What will the following code print?

```
char str1[10] = "Hi";
char str2[10];
strcpy(str2, str1);
printf("%s", str2);
```

a) Hi
b) str1
c) str2
d) Error

Answer: a) Hi

Q27. Which of the following is true about the `strcat()` function?
a) Concatenates two strings
b) Copies one string to another
c) Compares two strings
d) Finds the length of a string

Answer: a) Concatenates two strings

Q28. How can a 2D array be accessed in C?
a) Using single index
b) Using two indices `[row][column]`
c) Using pointer only
d) Cannot be accessed directly

Answer: b) Using two indices `[row][column]`

Q29. What is the null character in C strings?
a) '\n'
b) '\0'
c) '0'
d) NULL

Answer: b) '\0'

Q30. Which of the following is true about string arrays?
a) Strings are immutable
b) Strings are arrays of characters terminated by '\0'
c) Strings can store integers directly
d) Strings do not need memory allocation

Answer: b) Strings are arrays of characters terminated by '\0'

Q31. Which of the following functions **compares two strings** lexicographically?
a) strcmp()
b) strcat()
c) strcpy()
d) strlen()

Answer: a) strcmp()

Q32. What will the output of the following code be?

```
char str[10] = "Hello";
str[0] = 'h';
printf("%s", str);
```

a) Hello
b) hello
c) H
d) Error

Answer: b) hello

Q33. Which of the following is **true for a 1D array of size n**?
a) First element at index 1, last at n
b) First element at index 0, last at n-1

c) First element at index 0, last at n
d) First element at index 1, last at n-1

Answer: b) First element at index 0, last at n-1

Q34. Which built-in function **concatenates only part of a string** in C?
a) strncat()
b) strcat()
c) strcpy()
d) strncpy()

Answer: a) strncat()

Q35. Which of the following statements about 2D arrays is correct?
a) Memory is allocated row-wise
b) Memory is allocated column-wise
c) 2D arrays are dynamically sized by default
d) 2D arrays cannot be passed to functions

Answer: a) Memory is allocated row-wise

Q36. What is the value of `arr[0][1]` in the following 2D array?

```
int arr[2][2] = {{1,2},{3,4}};
```

a) 1
b) 2
c) 3
d) 4

Answer: b) 2

Q37. Which function **copies n characters from one string to another**?
a) strncpy()
b) strcpy()
c) strncat()
d) strcat()

Answer: a) strncpy()

Q38. What will be the output of the following code?

```
char str1[10] = "Hello";
```

```
char str2[10] = "World";
strcat(str1, str2);
printf("%s", str1);
```

a) Hello
b) World
c) HelloWorld
d) Error

Answer: c) HelloWorld

Q39. Which of the following is true about a string in C?
a) Can contain null character anywhere
b) Must end with null character `\0`
c) Cannot be stored in arrays
d) Length is always 10

Answer: b) Must end with null character `\0`

Q40. Which of the following correctly declares a 2D array of characters for storing 3 strings of maximum length 10?
a) `char str[3][10];`
b) `char str[10][3];`
c) `char str[3][];`
d) `char str[][10];`

Answer: a) `char str[3][10];`

UNIT 5: Functions in C

Q1. Which of the following best describes a function in C?
a) A block of code that performs a specific task and can be reused
b) A variable that stores values
c) A type of loop
d) A preprocessor directive

Answer: a) A block of code that performs a specific task and can be reused

Q2. Which of the following is a correct **function declaration**?
a) `int sum(int, int);`
b) `sum(int a, int b) { return a+b; }`
c) `int sum(a,b);`
d) `function sum(int, int);`

Answer: a) `int sum(int, int);`

Q3. What is the correct **function definition** for adding two integers?
a) `int add(int x, int y) { return x+y; }`
b) `add(int x, int y) { return x+y; }`
c) `int add(x, y) = x+y;`
d) `int add(x, y) { x+y; }`

Answer: a) `int add(int x, int y) { return x+y; }`

Q4. Which of the following **correctly calls** a function `sum` that returns an integer?
a) `sum(2,3);`
b) `int result = sum(2,3);`
c) `printf("%d", sum(2,3));`
d) All of the above

Answer: d) All of the above

Q5. Which type of **parameter passing** is used by default in C?
a) Call by reference
b) Call by value
c) Call by pointer
d) Call by address

Answer: b) Call by value

Q6. What is **recursion** in C?
a) A function that calls itself
b) A function that returns multiple values
c) A loop inside a function
d) A preprocessor directive

Answer: a) A function that calls itself

Q7. Which of the following statements about recursion is **true**?
a) Recursion can always be replaced by loops
b) Recursion always consumes less memory than loops
c) Recursion cannot call other functions
d) Recursion cannot be used in C

Answer: a) Recursion can always be replaced by loops

Q8. Which of the following is a **preprocessor directive** in C?
a) `#include`
b) `#define`
c) `#ifdef`
d) All of the above

Answer: d) All of the above

Q9. What is the purpose of **macros** in C?
a) To define constants or inline code snippets
b) To declare variables
c) To create loops
d) To call functions

Answer: a) To define constants or inline code snippets

Q10. Which of the following is **correct** for defining a macro?
a) `#define PI 3.14`
b) `#macro PI 3.14`
c) `#define PI = 3.14`
d) `macro PI = 3.14`

Answer: a) `#define PI 3.14`

Q11. Which of the following is **true about call by value**?
a) The actual parameter is changed
b) Only a copy of the value is passed
c) Memory address of the variable is passed
d) Cannot pass integers

Answer: b) Only a copy of the value is passed

Q12. What is the output of the following code?

```
void func(int x){
    x = x + 5;
}
int main(){
    int a = 10;
    func(a);
    printf("%d", a);
}
```

a) 15
b) 10
c) 5
d) Error

Answer: b) 10

Q13. Which of the following **cannot be used as a return type** for a function?
a) int
b) float
c) void
d) All can be used

Answer: d) All can be used

Q14. What is **recursion base case**?
a) The case where function calls itself
b) The condition that stops recursion
c) The first line of function
d) None of the above

Answer: b) The condition that stops recursion

Q15. Which of the following is a **valid recursive function** to calculate factorial?

a) `int fact(int n){ if(n==0) return 1; else return n*fact(n-1); }`
b) `int fact(int n){ return n*fact(n-1); }`
c) `int fact(int n){ if(n==0) return 0; else return n*fact(n-1); }`
d) `int fact(int n){ return n!; }`

Answer: a) `int fact(int n){ if(n==0) return 1; else return n*fact(n-1); }`

Q16. Which of the following **preprocessor directives** is used to prevent multiple inclusions of a header file?

a) #define
b) #ifndef … #endif
c) #include
d) #ifdef

Answer: b) #ifndef … #endif

Q17. What is the output of the following code?

```
int add(int a, int b){
    return a + b;
}
int main(){
    printf("%d", add(2,3));
}
```

a) 23
b) 5
c) 2
d) Error

Answer: b) 5

Q18. Which of the following is **true about macros**?

a) Evaluated at runtime
b) Evaluated at compile-time
c) Cannot take arguments
d) Cannot be used to define constants

Answer: b) Evaluated at compile-time

Q19. Which of the following is **true about recursion**?
a) Recursion always improves performance
b) Each recursive call uses stack memory
c) Recursive calls cannot call other functions
d) Recursion is mandatory in C

Answer: b) Each recursive call uses stack memory

Q20. Which of the following statements is **true about function declaration**?
a) Declares return type, name, and parameters
b) Must include function body
c) Cannot be used without definition
d) Only used for recursive functions

Answer: a) Declares return type, name, and parameters

Q21. What will be the output of the following recursive function?

```
void func(int n){
    if(n>0){
        printf("%d ", n);
        func(n-1);
    }
}
int main(){
    func(3);
}
```

a) 3 2 1
b) 1 2 3
c) 3 2 1 0
d) 0 1 2 3

Answer: a) 3 2 1

Q22. Which of the following is **true about recursive functions**?
a) Every recursion must have a base case
b) Recursion cannot call itself more than once
c) Recursion is slower than loops in all cases
d) Recursion does not use stack memory

Answer: a) Every recursion must have a base case

Q23. Which of the following **preprocessor directives** is used to include standard library files?
a) #include
b) #define
c) #macro
d) #ifdef

Answer: a) #include

Q24. What is the output of this code?

```
int square(int n){

    return n*n;

}

int main(){

    printf("%d", square(4));

}
```

a) 8
b) 16
c) 4
d) Error

Answer: b) 16

Q25. Which of the following **correctly defines a macro with arguments**?
a) `#define SQUARE(x) x*x`
b) `#define SQUARE x*x`
c) `#macro SQUARE(x) x*x`
d) `#define SQUARE(x) {x*x}`

Answer: a) `#define SQUARE(x) x*x`

Q26. What will the output of this code be?

```
int func(int a){
    if(a<=1) return 1;
    return a*func(a-1);
}
int main(){
    printf("%d", func(4));
}
```

a) 4
b) 10
c) 24
d) 16

Answer: c) 24

Q27. Which of the following is **true about call by value**?
a) Changes made in function reflect in actual parameter
b) Changes made in function do not affect the actual parameter
c) Can only be used for integers
d) Requires pointers

Answer: b) Changes made in function do not affect the actual parameter

Q28. What will be the output of the following code?

```
#define PI 3.14
int main(){
    printf("%f", PI);
}
```

a) 3.14
b) PI
c) Error
d) 0

Answer: a) 3.14

Q29. Which of the following statements is true about function **returning multiple values**?
a) C functions can return multiple values directly
b) Use pointers or structures to return multiple values
c) Use arrays to return multiple values
d) Both b and c

Answer: d) Both b and c

Q30. Which of the following **preprocessor directives** checks whether a macro is already defined?
a) #ifdef
b) #ifndef
c) #define
d) #include

Answer: a) #ifdef

Q31. What will be the output of the following code?

```
void func(int n){
    if(n==0) return;
    func(n-1);
    printf("%d ", n);
}
int main(){
    func(3);
}
```

a) 3 2 1
b) 1 2 3
c) 0 1 2 3
d) 3 2 1 0

Answer: b) 1 2 3

Q32. Which of the following is **true about preprocessor macros**?
a) Replaced by the compiler at runtime
b) Replaced by the preprocessor before compilation
c) Cannot take arguments
d) Only used for including files

Answer: b) Replaced by the preprocessor before compilation

Q33. Which of the following is **true for recursion**?
a) A function can recursively call itself indirectly through another function
b) Recursive calls cannot be nested
c) Recursion cannot have parameters
d) Recursive function always executes once

Answer: a) A function can recursively call itself indirectly through another function

Q34. Which of the following **correctly returns the sum of two numbers**?
a) `int sum(int a, int b){ return a+b; }`
b) `sum(int a, int b){ return a+b; }`
c) `int sum(a,b){ return a+b; }`
d) `void sum(int a, int b){ return a+b; }`

Answer: a) `int sum(int a, int b){ return a+b; }`

Q35. Which of the following **preprocessor directives** defines a symbolic constant?
a) #define
b) #include
c) #ifdef
d) #endif

Answer: a) #define

Q36. What will the output of the following code be?

```
#define SQUARE(x) x*x

int main(){

    printf("%d", SQUARE(3+1));

}
```

a) 16
b) 12
c) 9
d) 7

Answer: b) 12
*(Because macro expands to 3+13+1 = 3+3+1 = 12; demonstrates macro evaluation without parentheses.)**

Q37. Which of the following is **true about call by value**?
a) Changes to parameters affect the original argument
b) Only copies of variables are passed to the function
c) Cannot pass arrays
d) Requires recursion

Answer: b) Only copies of variables are passed to the function

Q38. What will be the output of this code?

```
int main(){

    #define X 5

    #undef X

    #define X 10

    printf("%d", X);

}
```

a) 5
b) 10
c) 0
d) Error

Answer: b) 10

Q39. Which of the following statements about recursion is **false**?
a) Each recursive call creates a new stack frame
b) Recursion can be replaced by iteration
c) Recursion always consumes less memory than loops
d) Recursion must have a base condition

Answer: c) Recursion always consumes less memory than loops

Q40. Which of the following is **true about function definition**?
a) Function definition provides actual body of the function
b) Function definition only declares the function
c) Function definition is optional if declared
d) Function definition cannot return values

Answer: a) Function definition provides actual body of the function

UNIT 6: Structures and Unions

Q1. Which of the following best defines a structure in C?
a) A collection of variables of the same data type
b) A collection of variables of different data types under a single name
c) A function that returns multiple values
d) A pointer to variables

Answer: b) A collection of variables of different data types under a single name

Q2. How do you declare a structure named `Student`?
a) `struct Student;`
b) `struct Student { int id; char name[20]; };`
c) `Student { int id; char name[20]; };`
d) `struct Student(int id, char name[20]);`

Answer: b) `struct Student { int id; char name[20]; };`

Q3. How do you access a member `id` of a structure variable `s`?
a) `s->id`
b) `s.id`
c) Both a and b depending on variable type
d) `s[id]`

Answer: c) Both a and b depending on variable type

Q4. Which of the following declares an **array of 10 structures** of type `Student`?
a) `struct Student s[10];`
b) `struct Student[10] s;`
c) `Student s[10];`
d) `struct s[10] Student;`

Answer: a) `struct Student s[10];`

Q5. Which of the following is **true about nested structures**?
a) A structure can contain another structure as a member
b) Structures cannot be nested
c) Nested structures cannot access members
d) Nested structures can only contain arrays

Answer: a) A structure can contain another structure as a member

Q6. Which of the following correctly declares a union named `Data`?

a) `union Data { int i; float f; char str[20]; };`
b) `union Data;`
c) `Data union { int i; float f; };`
d) `union { int i; float f; } Data;`

Answer: a) `union Data { int i; float f; char str[20]; };`

Q7. Which of the following is **true about unions**?

a) All members share the same memory location
b) Each member has a separate memory location
c) Unions cannot contain different data types
d) Unions cannot be nested

Answer: a) All members share the same memory location

Q8. What is the key difference between a structure and a union?

a) Structure stores all members at the same memory location, union stores separately
b) Structure stores each member separately, union shares memory among all members
c) Structure can only have same data types, union can have different types
d) No difference

Answer: b) Structure stores each member separately, union shares memory among all members

Q9. Which operator is used to access members of a structure through a pointer?

a) `.`
b) `->`
c) `*`
d) `&`

Answer: b) `->`

Q10. Which of the following statements about arrays of structures is **true**?

a) All elements are stored contiguously in memory
b) Only the first element is stored
c) Array of structures cannot be initialized
d) Each element can only hold one member

Answer: a) All elements are stored contiguously in memory

Q11. What will be the output of the following code?

```
struct Point { int x; int y; };

struct Point p1 = {1, 2};

printf("%d", p1.y);
```

a) 1
b) 2
c) 0
d) Error

Answer: b) 2

Q12. Which of the following is **true about nested structures**?
a) Only one level of nesting is allowed
b) Nested structures can be accessed using the dot operator
c) Nested structures cannot be arrays
d) Nested structures cannot be initialized

Answer: b) Nested structures can be accessed using the dot operator

Q13. How do you access a member `f` of a union `Data` using a pointer `ptr`?
a) `ptr.f`
b) `ptr->f`
c) `(*ptr).f`
d) Both b and c

Answer: d) Both b and c

Q14. Which of the following statements about unions is **false**?
a) Only one member can store a value at a time
b) All members share the same memory location
c) Union size is equal to the sum of sizes of all members
d) Union can contain different data types

Answer: c) Union size is equal to the sum of sizes of all members

Q15. Which of the following is **true about structures**?
a) Each member has its own memory location
b) Members share the same memory

c) Cannot contain arrays
d) Cannot be passed to functions

Answer: a) Each member has its own memory location

Q16. How do you declare a pointer to a structure `Student`?

a) `struct Student *ptr;`
b) `Student ptr*;`
c) `struct *Student ptr;`
d) `Student *ptr;`

Answer: a) `struct Student *ptr;`

Q17. Which of the following is **true about initializing an array of structures**?

a) `struct Student s[2] = {{1,"Alice"},{2,"Bob"}};`
b) `struct Student s[2] = {1,"Alice",2,"Bob"};`
c) `struct Student s[2]; s = {{1,"Alice"},{2,"Bob"}};`
d) Cannot initialize

Answer: a) `struct Student s[2] = {{1,"Alice"},{2,"Bob"}};`

Q18. Which operator is used to access members of a structure?

a) `.`
b) `->`
c) `*`
d) `&`

Answer: a) `.`

Q19. What is the output of the following union code?

```
union Data { int i; char c; };

union Data d;

d.i = 65;

printf("%c", d.c);
```

a) A
b) 65
c) Error
d) Depends on compiler

Answer: d) Depends on compiler
(Accessing a different member of a union may result in implementation-dependent value.)

Q20. Which of the following is a **key difference between structure and union**?
a) Union members cannot be pointers
b) Structure members occupy separate memory, union members share memory
c) Structure cannot have nested structures
d) Union cannot store integers

Answer: b) Structure members occupy separate memory, union members share memory

Q21. Which of the following statements about **arrays of structures** is true?
a) All elements can be accessed using an index and dot operator
b) Only the first element can be accessed
c) Arrays of structures cannot be initialized
d) Elements cannot store different data types

Answer: a) All elements can be accessed using an index and dot operator

Q22. Which of the following is **true about nested structures**?
a) They can contain other structures as members
b) They cannot be passed to functions
c) Only arrays can be nested
d) Nested structures cannot have pointers

Answer: a) They can contain other structures as members

Q23. What is the size of the following union on a system where `int` is 4 bytes and `char` is 1 byte?

```
union Data { int i; char c; };
```

a) 1
b) 4
c) 5
d) 8

Answer: b) 4
(Union size is determined by the largest member.)

Q24. Which of the following **cannot be done with unions**?
a) Store different data types
b) Access multiple members simultaneously
c) Store one member at a time
d) Assign values to members

Answer: b) Access multiple members simultaneously

Q25. How do you **pass a structure to a function**?
a) By value
b) By reference (using pointers)
c) Both a and b
d) Cannot pass structures

Answer: c) Both a and b

Q26. Which of the following is true about the **dot (.) operator**?
a) Used to access members of a structure variable
b) Used to access members through a pointer
c) Used to declare structures
d) Used to assign values to arrays

Answer: a) Used to access members of a structure variable

Q27. What will the output of the following code be?

```
struct Point { int x; int y; };

struct Point p1 = {1,2};

struct Point *ptr = &p1;

printf("%d", ptr->x);
```

a) 1
b) 2
c) Error
d) Garbage value

Answer: a) 1

Q28. Which of the following is **true about union memory allocation**?

a) All members share the same memory location

b) Each member has its own memory

c) Unions cannot store different data types

d) Union size is always 1 byte

Answer: a) All members share the same memory location

Q29. Which of the following **correctly declares a nested structure**?

a) `struct Date { int day; int month; int year; }; struct Student { char name[20]; struct Date dob; };`

b) `struct Student { char name[20]; int Date { day, month, year }; };`

c) `struct Student { char name[20]; Date dob; };`

d) `struct Student { char name[20]; struct dob { int day, month, year; }; };`

Answer: a) `struct Date { int day; int month; int year; }; struct Student { char name[20]; struct Date dob; };`

Q30. Which of the following statements is **true about structure and union differences**?

a) Structures share memory, unions allocate separate memory for each member

b) Structures allocate separate memory for each member, unions share memory

c) Both structures and unions share memory

d) Both structures and unions allocate separate memory for each member

Answer: b) Structures allocate separate memory for each member, unions share memory

ONE LINE QUESTIONS

UNIT 1: Introduction to Computers and Programming Concepts

Q1. Define a computer.
Answer: A computer is an electronic device that accepts input, processes data, and produces output.

Q2. What is hardware?
Answer: Hardware is the physical components of a computer system.

Q3. What is software?
Answer: Software is a set of programs or instructions that tell the computer what to do.

Q4. Differentiate system software and application software.
Answer: System software manages hardware and provides a platform; application software performs specific tasks for users.

Q5. Define an operating system.
Answer: An operating system is system software that manages computer hardware and software resources.

Q6. What is a compiler?
Answer: A compiler translates high-level programming code into machine code before execution.

Q7. What is an interpreter?
Answer: An interpreter executes high-level code line by line without producing a separate machine code file.

Q8. Define high-level programming language.
Answer: High-level languages are close to human language and easier to program, e.g., C, Python.

Q9. Define assembly-level language.
Answer: Assembly language is a low-level language using mnemonics to represent machine instructions.

Q10. Define machine-level language.
Answer: Machine-level language consists of binary instructions understood directly by the CPU.

Q11. What is an algorithm?
Answer: An algorithm is a step-by-step procedure to solve a problem.

Q12. What is a flowchart?
Answer: A flowchart is a graphical representation of an algorithm.

Q13. Name two types of software.
Answer: System software and application software.

Q14. Give an example of system software.
Answer: Windows OS or Linux.

Q15. Give an example of application software.
Answer: Microsoft Word or Excel.

Q16. What is the main function of the CPU?
Answer: To process data and execute instructions.

Q17. What is input device?
Answer: A device used to enter data into a computer, e.g., keyboard, mouse.

Q18. What is output device?
Answer: A device that displays or produces the processed information, e.g., monitor, printer.

Q19. Define memory in a computer.
Answer: Memory is the storage area for data and instructions in a computer.

Q20. Differentiate primary and secondary memory.
Answer: Primary memory (RAM) is temporary and fast; secondary memory (hard disk) is permanent and slower.

UNIT 2: Introduction to C Language and Program Structure

Q1. Name two features of C language.
Answer: Simple, portable, structured, and high-level language.

Q2. What is a header file in C?
Answer: A header file contains definitions and function declarations to be included in a C program.

Q3. How are comments written in C?
Answer: Single-line comments use `//`, multi-line comments use `/* ... */`.

Q4. Define a variable in C.
Answer: A variable is a named memory location that stores data which can be modified during program execution.

Q5. Define a constant in C.
Answer: A constant is a memory location whose value cannot be changed during program execution.

Q6. Name four basic data types in C.
Answer: int, float, char, double.

Q7. Define an operator in C.
Answer: An operator is a symbol that tells the compiler to perform specific operations on data.

Q8. What is an expression in C?
Answer: An expression is a combination of variables, constants, and operators that evaluates to a value.

Q9. Define type conversion in C.
Answer: Type conversion is the process of converting a value from one data type to another.

Q10. What is operator precedence?
Answer: Operator precedence determines the order in which operators are evaluated in an expression.

Q11. Define a constant literal in C.
Answer: A constant literal is a fixed value written directly in the code, e.g., 10, 3.14, 'A'.

Q12. Define a character constant.
Answer: A character constant represents a single character enclosed in single quotes, e.g., 'A'.

Q13. Define an integer constant.
Answer: An integer constant is a numeric value without a decimal point, e.g., 100.

Q14. Define a floating-point constant.
Answer: A floating-point constant is a numeric value with a decimal point, e.g., 3.14.

Q15. What is a literal string in C?
Answer: A literal string is a sequence of characters enclosed in double quotes, e.g., "Hello".

Q16. What is the role of `main()` function in C?
Answer: `main()` is the entry point of a C program where execution begins.

Q17. What is the difference between declaration and definition of a variable?
Answer: Declaration informs the compiler about the variable type; definition allocates memory for it.

Q18. What is an identifier in C?
Answer: An identifier is a name used to identify a variable, function, or structure in C.

Q19. What is the difference between signed and unsigned data types?
Answer: Signed data types can store positive and negative values; unsigned types store only positive values.

Q20. What is the purpose of `#include` directive?
Answer: `#include` is used to include header files in a C program.

UNIT 3: Control Structures in C

Q1. What is an if statement?
Answer: An if statement executes a block of code when a condition is true.

Q2. Differentiate if and if-else statements.
Answer: If executes code when the condition is true; if-else executes one block if true, another if false.

Q3. Define a nested if statement.
Answer: A nested if is an if statement placed inside another if or else block.

Q4. What is a switch-case statement used for?
Answer: Switch-case executes one block among multiple options based on a variable's value.

Q5. What is a loop?
Answer: A loop is a control structure that repeats a block of code while a condition is true.

Q6. Name three types of loops in C.
Answer: for, while, do-while.

Q7. Define a for loop.
Answer: A for loop repeatedly executes a block of code for a known number of iterations.

Q8. Define a while loop.
Answer: A while loop executes a block of code as long as the condition is true.

Q9. Define a do-while loop.
Answer: A do-while loop executes a block of code at least once and then repeats while the condition is true.

Q10. What is the use of the break statement?
Answer: Break immediately terminates a loop or switch-case.

Q11. What is the use of the continue statement?
Answer: Continue skips the current iteration of a loop and moves to the next iteration.

Q12. What is a conditional expression?
Answer: A conditional expression evaluates to true or false and controls decision-making.

Q13. What is an infinite loop?
Answer: An infinite loop is a loop that never terminates because the condition is always true.

Q14. Give an example of a loop with a known number of iterations.
Answer: for(i=0; i<10; i++) { /* code */ }

Q15. Give an example of a loop with an unknown number of iterations.
Answer: while(flag) { /* code */ }

Q16. Can a switch statement use float or double as an expression?
Answer: No, switch expression must be an integer or char type.

Q17. Can a break statement be used outside loops or switch?
Answer: No, break can only be used inside loops or switch-case.

Q18. Can a continue statement be used outside loops?
Answer: No, continue is only valid inside loops.

Q19. What happens if the condition in a do-while loop is false initially?
Answer: The loop body executes once before checking the condition.

Q20. Can loops be nested in C?
Answer: Yes, loops can be nested inside other loops.

UNIT 4: Arrays and Strings

Q1. Define an array.
Answer: An array is a collection of elements of the same data type stored in contiguous memory.

Q2. What is a one-dimensional array?
Answer: A one-dimensional array is a single row of elements of the same type.

Q3. What is a two-dimensional array?
Answer: A two-dimensional array is an array of arrays, like a matrix.

Q4. How do you declare a one-dimensional array of 10 integers?
Answer: `int arr[10];`

Q5. How do you initialize an array at the time of declaration?
Answer: `int arr[3] = {1, 2, 3};`

Q6. How do you access the 3rd element of an array `arr`?
Answer: `arr[2];`

Q7. Define a string in C.
Answer: A string is an array of characters terminated by a null character `\0`.

Q8. How is a string stored in memory?
Answer: As a contiguous array of characters with a `\0` at the end.

Q9. Name a function to find the length of a string.
Answer: `strlen()`

Q10. Name a function to copy one string to another.
Answer: `strcpy()`

Q11. Name a function to concatenate two strings.
Answer: `strcat()`

Q12. What is the null character in C?
Answer: `\0` marks the end of a string.

Q13. Can arrays be passed to functions in C?
Answer: Yes, arrays can be passed to functions by reference.

Q14. Can strings be passed to functions in C?
Answer: Yes, strings are passed as character arrays.

Q15. Define an array of structures.
Answer: An array of structures is a collection of structure variables stored consecutively.

Q16. Can arrays have negative indices in C?
Answer: No, array indices must be zero or positive integers.

Q17. What is the index of the first element in an array?
Answer: 0

Q18. What is the index of the last element in an array of size n?
Answer: n-1

Q19. Can two-dimensional arrays be passed to functions?
Answer: Yes, by specifying all but the first dimension.

Q20. How do you declare a string of 20 characters?
Answer: `char str[20];`

UNIT 5: Functions in C

Q1. Define a function in C.
Answer: A function is a block of code that performs a specific task and can be called multiple times.

Q2. What is the difference between function declaration and definition?
Answer: Declaration informs the compiler about the function name and parameters; definition provides the actual code body.

Q3. What is a function call?
Answer: A function call executes the code of a function.

Q4. What is the return value of a function?
Answer: The value that a function sends back to the caller after execution.

Q5. What is call by value?
Answer: A method where a copy of the actual argument is passed to the function.

Q6. What is call by reference?
Answer: A method where the address of the actual argument is passed, allowing the function to modify the original variable.

Q7. Define recursion.
Answer: Recursion is when a function calls itself either directly or indirectly.

Q8. What is a base case in recursion?
Answer: The condition that stops the recursive calls.

Q9. Can functions return multiple values in C?
Answer: No, a function can return only one value; multiple values can be returned using structures or pointers.

Q10. What is a preprocessor directive?
Answer: A command that is processed before compilation, e.g., `#define` or `#include`.

Q11. What is a macro in C?
Answer: A macro is a preprocessor directive that defines a symbolic name or code snippet.

Q12. Give an example of defining a macro.
Answer: `#define PI 3.14`

Q13. What is a void function?
Answer: A function that does not return any value.

Q14. Can a function call itself indirectly?
Answer: Yes, this is called indirect recursion.

Q15. Can a function call another function?
Answer: Yes, a function can call any other function.

Q16. What is the purpose of the `return` statement?
Answer: To send a value back to the calling function and terminate the function.

Q17. Can a function have no parameters?
Answer: Yes, a function can have zero parameters.

Q18. Can a function be defined inside another function in C?
Answer: No, functions cannot be nested in C.

Q19. What is the role of the `main()` function?
Answer: `main()` is the entry point of a C program.

Q20. Can a function be recursive without a base case?
Answer: Technically yes, but it will cause infinite recursion and crash the program.

UNIT 6: Structures and Unions

Q1. Define a structure in C.
Answer: A structure is a user-defined data type that groups variables of different data types under a single name.

Q2. How do you declare a structure named `Student`?
Answer: `struct Student { int id; char name[20]; };`

Q3. How do you access a member `id` of a structure variable `s`?
Answer: `s.id`

Q4. Define a nested structure.
Answer: A nested structure is a structure that contains another structure as a member.

Q5. Define an array of structures.
Answer: An array of structures is a collection of structure variables stored consecutively in memory.

Q6. Define a union in C.
Answer: A union is a user-defined data type where all members share the same memory location.

Q7. What is the key difference between a structure and a union?
Answer: Structures allocate separate memory for each member; unions share the same memory for all members.

Q8. Which operator is used to access structure members through a pointer?
Answer: `->`

Q9. Can a union store multiple values simultaneously?
Answer: No, a union can store only one value at a time.

Q10. Can structures be passed to functions in C?
Answer: Yes, by value or by reference using pointers.

Q11. Can unions be passed to functions in C?
Answer: Yes, similar to structures, by value or by reference.

Q12. What is the size of a union determined by?
Answer: The size of the largest member of the union.

Q13. Can structures contain arrays as members?
Answer: Yes, structures can have arrays as members.

Q14. Can unions contain arrays as members?
Answer: Yes, unions can have arrays as members.

Q15. Can structures be nested multiple levels?
Answer: Yes, structures can be nested to any level.

Q16. How do you initialize a structure at declaration?
Answer: `struct Student s1 = {1, "Alice"};`

Q17. How do you declare a pointer to a structure?
Answer: `struct Student *ptr;`

Q18. How do you access a member using a structure pointer?
Answer: `ptr->id` or `(*ptr).id`

Q19. What is one advantage of using structures?
Answer: Structures allow grouping of related data of different types under a single name.

Q20. Can unions contain structures as members?
Answer: Yes, unions can contain structures as members.