1. Introduction to C Programming

IMP Questions + Answers

1. What is C language?

C is a **procedural, general-purpose programming language** developed by Dennis Ritchie in 1972 at Bell Labs.

2. History of C language?

Developed from **BCPL → B → C** by Dennis Ritchie (1972). Standardized by **ANSI in 1989 (ANSI C)**.

3. Applications of C?

Operating systems, embedded systems, compilers, device drivers, databases.

4. Features of C?

Fast, portable, mid-level, structured, rich library, supports pointers, modular.

5. Structure of a C Program?

Header files → main() → declarations → statements → return.

6. Execution flow of C program?

Edit → Compile → Link → Load → Execute.

7. Reading a character?

Using getchar() or scanf("%c", &ch);

8. Writing a character?

Using putchar(ch); or printf("%c", ch);

9. Formatted Input functions?

scanf()

10. Formatted Output functions?

printf()

Short Notes

Structure of C Program

```
#include <stdio.h>


int main() {

   // code

   return 0;

}
```

---

2. Constants, Variables, Data Types, Operators

IMP Questions + Answers

1. What are constants?

Fixed values that do not change during execution.
Example: 10, 3.14, 'A'

2. What are variables?

Names given to memory locations whose value can change.

3. What are keywords?

Reserved words like int, float, return.

4. What is an identifier?

User-defined name for variables/functions.

5. What is typedef?

Gives an alias to a data type.
Example:

```
typedef int integer;
```

6. What are symbolic constants?

Created using #define.

```
#define PI 3.14
```

7. Types of Operators?

Arithmetic, Relational, Logical, Increment/Decrement, Assignment, Conditional (?:), Bitwise.

8. Precedence of Operators?

() → ++/-- → *, /, % → +, - → <, > → == → && → || → =

9. Type conversion?

Automatic (implicit) and manual (explicit casting).

Short Notes

Variable Declaration

int a;

float b;

char c;

Assignment

a = 10;

Operator Example

int c = a + b * 2;

---

3. Control Structures in C

IMP Questions + Answers

1. What is decision making?

Choosing a block of code based on a condition.

2. Syntax of if statement?

if (condition) { }

3. Nested if-else?

if inside another if.

4. Switch statement?

Used when comparing one variable with multiple values.

5. Break statement?

Used to exit loop or switch.

6. Ternary operator?

Short form of if-else:

```
(condition ? expr1 : expr2)
```

7. While loop syntax?

```
while (condition) { }
```

8. Do-while loop?

Executes once before checking condition.

9. For loop syntax?

```
for (i=0; i<n; i++) { }
```

10. goto statement?

Jumps to a labeled statement.

Short Notes

If Example

```
if (a > b) {
   printf("A is greater");
}
```

Switch Example

```
switch(day){
   case 1: printf("Mon"); break;
}
```

Loop Example

```
for(int i=0; i<5; i++)
   printf("%d", i);
```

---

4. Arrays and Strings

IMP Questions + Answers

1. What is an array?

Collection of similar data elements stored in contiguous memory.

2. What is a 1-D array?

A linear list of elements.

3. What is a 2-D array?

Matrix-like structure with rows and columns.

4. String?

Character array ending with null '\0'.

5. Built-in string functions?

strlen(), strcpy(), strcmp(), strcat().

Short Notes

1D Array

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

2D Array

```
int mat[2][3];
```

String Declaration

```
char name[20] = "Shriram";
```

String Functions

```
strlen(name);

strcpy(dest, src);
```

5. Functions, Structures, Unions

IMP Questions + Answers

1. What is a user-defined function?

Function created by the user to perform a specific task.

2. Function prototype?

Declaration of function before use.

3. Function definition?

Actual body of the function.

4. Parameter passing methods?

Call by value & call by reference.

5. Recursion?

Function calling itself.

6. Macro?

Preprocessor substitution:

```
#define max 100
```

7. What is a structure?

Collection of variables of different data types under one name.

8. What is a union?

Similar to structures, but shares the same memory for all members.

9. Arrays of structures?

Multiple structure variables in an array.

Short Notes

Function Syntax

```
return_type function_name(parameters);
```

```
int add(int a, int b) {
   return a + b;
}
```

Call by Value

```
fun(a);
```

Recursive Example

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

Structure Syntax

```
struct student {
   int roll;
   char name[20];
};

struct student s1;
```

Union Syntax

```
union data {
   int i;
   float f;
   char c;
};
```