UnCodemy

C Programming (PPS) Important Questions PDF

Programming for Problem Solving in C usually has five units: computer basics and algorithms, operators and conditional statements, loops and arrays, functions and recursion, and pointers, structures and file handling. Most marks come from writing short C programs (prime, palindrome, sorting, matrix, string handling), output-prediction questions and theory on storage classes, pointers and dynamic memory.

Key takeaways

- Unit 1: components of a computer, algorithms, flowcharts, compilation steps, data types and operator precedence.
- Units 2 and 3: if-else, switch, loops, arrays, strings, searching and sorting programs.
- Unit 4: functions, call by value and by reference, recursion and storage classes.
- Unit 5: pointers, dynamic memory, structures and unions, file handling and preprocessor directives.
- These are practice questions based on common syllabus topics, not verified past papers.

About these important questions

These are **practice questions** built from topics that appear in most Indian B.Tech syllabi for Programming for Problem Solving (C), including AKTU and universities that follow the AICTE model curriculum. They are **not verified past papers**, and we do not claim that any question will appear in your exam. Unit numbers and topic order differ between universities, so first match each unit below with your own university syllabus. For real previous papers, use your university's official archive. AKTU students can download them from the AKTU old question paper page.

Each unit lists key definitions, derivations or theory questions, numericals with short answer outlines, and long-answer questions. Use the outlines to check your method, then write full solutions yourself.

Unit 1: Computer Basics, Algorithms and C Fundamentals

Key definitions: components of a computer (CPU, memory, I/O), system and application software, compiler, interpreter, assembler, linker, loader, algorithm, flowchart, pseudocode, tokens, keywords, identifiers, data types.

Important questions:

- Draw the block diagram of a computer and explain each unit.

- Explain the steps from writing a C program to running it: editing, preprocessing, compiling, linking, loading and executing.
- Write an algorithm and draw a flowchart to find the largest of three numbers.
- Explain the basic data types in C with their typical sizes and format specifiers. Note that sizes depend on the compiler and platform.
- Explain operator precedence and associativity with an example.

Output questions (answer outline):

- `int x = 10 / 3 * 3;` gives x = 9, because integer division happens first (10 / 3 = 3) and then 3 * 3.
- `printf("%d", 5 > 3 && 2 > 4);` prints 0, because the second condition is false.

Unit 2: Conditional Branching and Loops

Key definitions: if, if-else, nested if, else-if ladder, switch-case, while, do-while, for, break, continue, goto.

Programs to practise:

- Check whether a number is prime.
- Check whether a number is an Armstrong number or a palindrome.
- Print the Fibonacci series up to n terms.
- Find the factorial of a number using a loop.
- Print a number pyramid or star pattern.
- Make a simple calculator using switch-case.

Long-answer questions: Compare while and do-while with an example where the output is different. Explain break and continue with programs.

A model prime-check program:

```
#include <stdio.h>
int main(void) {
    int n, i, isPrime = 1;
    scanf("%d", &n);
    if (n < 2) isPrime = 0;
    for (i = 2; i * i <= n; i++)
        if (n % i == 0) { isPrime = 0; break; }
    printf(isPrime ? "Prime\n" : "Not prime\n");
    return 0;
}
```

Unit 3: Arrays and Strings

Key definitions: one-dimensional and two-dimensional arrays, array initialisation, strings as character arrays, null character, string library functions (strlen, strcpy, strcat, strcmp).

Programs to practise:

- Find the largest and smallest elements of an array.

- Sort an array using bubble sort and selection sort. Bubble sort makes about $n^2/2$ comparisons in the worst case.
- Search an element using linear search and binary search (binary search needs a sorted array).
- Add, multiply and transpose two matrices.
- Reverse a string and check whether it is a palindrome without using library functions.
- Count vowels, consonants and words in a string.

Long-answer questions: Explain how a 2D array is stored in memory (row-major order). Write a program for matrix multiplication and explain the condition on the orders.

Unit 4: Functions and Recursion

Key definitions: function declaration (prototype), definition and call, actual and formal parameters, call by value, call by reference, recursion, base case, storage classes (auto, register, static, extern).

Important questions:

- Explain call by value and call by reference with a swap program.
- Write recursive functions for factorial, Fibonacci, sum of digits and GCD.
- Tower of Hanoi: write the recursive solution. For n discs it needs $2^n - 1$ moves, so 3 discs need 7 moves.
- Explain the four storage classes with scope, lifetime and default value.
- Compare recursion and iteration.

Output question: a function with a `static int count` called three times prints 1, 2, 3, because a static local variable keeps its value between calls.

Unit 5: Pointers, Structures and File Handling

Key definitions: pointer, address-of and dereference operators, pointer arithmetic, pointer to pointer, array of pointers, dynamic memory allocation (malloc, calloc, realloc, free), structure, union, nested structure, file modes, command-line arguments, macros.

Important questions:

- Explain pointer arithmetic with an example on an integer array.
- Compare malloc and calloc. (calloc sets memory to zero and takes two arguments.)
- Write a program to allocate an array dynamically, read n numbers and print their sum.
- Compare structure and union. A union's size is at least the size of its largest member, while a structure's size is at least the sum of its members (padding can add bytes).
- Write a program using an array of structures to store and print student records.
- Write a program to copy one file to another using fopen, fgetc, fputc and fclose. Explain file modes r, w, a, r+, w+ and a+.
- Explain #define macros and the difference between a macro and a function.

How to score well in Programming for Problem Solving

- Write complete programs with header files, main, variable declarations and a sample output. Partial code loses marks.
- Add a short comment or one line explaining the logic before the code.
- Dry-run your program with one input in the answer sheet. It shows the examiner that the logic works.
- Practise output-prediction questions on operators, loops, static variables and pointers.
- Type and run each program at least once on a compiler before the exam.

Short-answer (2-mark) questions for Programming for Problem Solving (C)

Short questions test definitions and one-line results. Practise answering each in two to four lines:

1. What is the difference between a compiler and an interpreter?
2. Define an algorithm and list its properties.
3. What is the size of char in C? (Answer: 1 byte by definition.)
4. Differentiate between = and == in C.
5. What is the difference between break and continue?
6. Why is the null character needed at the end of a string?
7. What does strcmp return when the two strings are equal? (Answer: 0.)
8. What is a function prototype?
9. What is the default value of a static variable? (Answer: zero.)
10. What is a NULL pointer?
11. What is a dangling pointer?
12. What does the free() function do?

A 7-day revision plan for Programming for Problem Solving (C)

Use this plan in the last week before the exam. Adjust it to the units and weightage in your own syllabus.

- **Day 1 (Unit 1, Computer Basics, Algorithms and C Fundamentals):** revise the definitions, then solve the numericals and one long answer from this unit without notes.
- **Day 2 (Unit 2, Conditional Branching and Loops):** revise the definitions, then solve the numericals and one long answer from this unit without notes.
- **Day 3 (Unit 3, Arrays and Strings):** revise the definitions, then solve the numericals and one long answer from this unit without notes.
- **Day 4 (Unit 4, Functions and Recursion):** revise the definitions, then solve the numericals and one long answer from this unit without notes.
- **Day 5 (Unit 5, Pointers, Structures and File Handling):** revise the definitions, then solve the numericals and one long answer from this unit without notes.

- **Day 6:** attempt one full three-hour practice paper made from these questions, or an official previous paper from your university archive.
- **Day 7:** go through your formula sheet, the short-answer list and every question you got wrong in the practice paper.

Frequently asked questions

Which programs are most important for the C programming exam?

Prime number, factorial, Fibonacci, palindrome, Armstrong number, bubble sort, binary search, matrix multiplication, string reverse, swap using pointers and file copy are the programs most students should practise.

How should I write a program in the answer sheet?

Write the header files, main function, declarations, input, logic and output in order, add a one-line explanation, and show a sample input and output. Indent the code so it is easy to read.

Are pointers difficult to score in?

Pointers feel hard at first, but exam questions are usually standard: pointer arithmetic, swap by reference, dynamic arrays and pointer to structure. Drawing memory boxes with addresses makes them much easier.

Are these questions from real past papers?

No. These are practice questions based on topics that are common in B.Tech syllabi. They are not verified past papers, and no question is promised to appear in your exam. Check your own university syllabus and official previous papers; AKTU students can use the AKTU old question paper archive.

Is this PDF free?

Yes. Fill in the short form on this page and the PDF downloads right away.

Sources and further reading

- AKTU: Old question papers (official archive)
- AKTU: Syllabus
- NPTEL: Free online engineering courses
- AICTE: All India Council for Technical Education

About this PDF

Prepared by the Uncodemy academic team from common B.Tech syllabus topics. These are practice questions, not past papers. Corrections: info@uncodemy.com.

More free resources and courses: uncodemy.com | info@uncodemy.com | +91 98183 66550