Chhatrapati Shahu Ji Maharaj University, Kanpur

Answer Script Details
Barcode 7464907

Roll No. 23071002365
Total Mark 63/75.00

Exam BACHELOR OF COMPUTER APPLICATION_DEC-2023
Subject BCA1002 - II C PROGRAMMING

Question wise Mark Summary

Q.No	Mark	Q.No	Mark	Q.No	Mark	Q.No	Mark
1A	5/5	9	NA/15				
1B	4/5						
1C	4/5						
1D	5/5						
1E	5/5						
1F	4/5						
1G	4/5						
1H	4/5						
1I	5/5						
2	10/15						
3A	NA/7						
3B	NA/7						
4	NA/15						
5	NA/15						
6	NA/15						
7	13/15						
8	NA/15						

Chhatrapati Shahu Ji Maharaj University
Kanpur, Uttar Pradesh

PART-I

Date of Exam: 11/12/23 Shift: Morning Room No.: G-03

Paper Code: BCA-1002 Subject: C Prog. Year/Sem.: V/I

Name of Candidate: Khyati Trivedi

Roll No. 2307100236 5

Signature of Candidate: Khyati Trivedi

Signature of Invigilator: 11/12/23

COE Facsimile

PART-II

MARKS OBTAINED										
Q.	1	2	3	4	5	6	7	8	9	10
(a)										
(b)										
(c)										
(d)										
(e)										
(f)										
(g)										
(h)										
(i)										
(j)										
Total										
Total Marks in Figures									Max. Marks	
Total Marks in Words										

BCA1002

Paper Code

Signature of Evaluator

PART-III

Course BCA

Session 2023-2024 Year/ Semester I

Subject Name C Programming

Medium English ● Hindi ○

Paper Code: BCA1002

Exam Date: 11 12 2023

Name of Candidate: KHYATI TRIVEDI

Father's Name: S K TRIVEDI

महाविद्यालय का कोड College Code: KN162

परीक्षा केन्द्र का कोड Exam Centre Code: KN162

परीक्षा का प्रकार Type of Exam

Regular ● नियमित परीक्षार्थी Ex- Student ○

Private ○ बैक पेपर परीक्षा Back Paper Exam ○

ANSWER BOOKLET NO.

7464907

BCA1002

Paper Code

PART-IV

नामांकन संख्या Enrollment Number: CSJMA2300012951 7

अभ्यर्थी अनुक्रमांक संख्या Candidate's Roll Number: 2307100236 5

पेपर कोड Paper Code: BCA1002

Signature of Candidate

11/12/23

Signature of Invigilator

C S Facsimile

COE Facsimile

नोट:- 1. परीक्षार्थी को निर्देशित किया जाता है कि आवरण पत्रे के पृष्ठ भाग पर अंकित सभी निर्देशों को सावधानी पूर्वक पढ़ें।
2. बॉक्स में भरी जाने वाली प्रविष्टियाँ स्वयं से शुरू की जायें। 3. गोलों को काले या नीले बॉलपेन से भरा जाये।

INSTRUCTION TO THE CANDIDATE FOR FILLING PART-I

1. Read the instructions carefully given on the answer script and admit card.
2. Write Date of Exam, Shift, Paper Code & Name of Subject Correctly.
3. Write Name & Roll No. Correctly.
4. Write Semester & Branch Correctly.

INSTRUCTION TO THE CANDIDATE FOR FILLING PART-III

1. Use blue or black ball point pen for writing alphabets & numerals in ☐ boxes.
2. Carefully study the example before you start marking.
3. As shown in the example below, blacken the circles completely.

4. Make no Stray marks on this sheet.
5. **DO NOT WRITE OR MARK ON THE BAR CODE.**

IN ORDER TO AVOD UFM (UNFAIR MEANS) :

1. The Roll No. and Answer Book no. found elsewhere or any other symbol found in the answer book will be treated as unfair means.
2. Any tempering of Bar Code and Booklet no shall be treated as Unfair Means.
3. Do Not bring the materials like slip of paper/mobile/digital diaries/ study material/ revision notes in examination hall. Possession of the mobiles/ digital diaries/electronic/digital/ watch and any other electronic gadget except memory less scientific calculator shall be considered as UFM case.
4. Do not keep or paste currency note in answer script it shall be consider as UFM.

अनुचित साधन से बचने हेतु :

1. उत्तर पुस्तिका के निर्देशित स्थान को छोड़कर अनुक्रमांक एवं उत्तरपुस्तिका का क्रमांक कहीं और न लिखे तथा कोई भी चिन्ह न बनायें क्योंकि यह अनुचित साधन प्रयोग की परिधि में आता है।
2. उत्तर पुस्तिका से बारकोड अथवा उत्तर पुस्तिका संख्या पर छेड़ छाड़ करने पर अनुचित साधन प्रयोग माना जायेगा।
3. परीक्षा कक्ष में किसी अनुचित साधन न लायें, जैसे लिखे हुए कागज के टुकड़ें, मोबाईल, डिजिटल डायरी, डिजिटल घड़ी, कॉपी, पुस्तक यह सभी वस्तुएं जो अनुचित साधन के अन्तर्गत आती है। केवल संशोधित प्रश्नपत्र में ही मेमोरी लेस साइंटिफिक कैलकुलेटर ले जाने की अनुमति होगी।
4. उत्तर पुस्तिकाओं में कागज न रखें न ही उत्तर पुस्तिका में लिखायें। ऐसा करना अनुचित साधन प्रयोग की परिधि में आता है।

परीक्षार्थियों के लिए निर्देश

1. प्रवेश पत्र एवं उत्तर पुस्तिका पर दिये गये निर्देशों को ध्यान से पढ़ें।
2. उत्तर पृष्ठ के दूसरी तरफ कुछ न लिखें।
3. उत्तर पुस्तिका के पृष्ठों पर दोनों तरफ लिखें।
4. प्रश्न पत्र पर अपने अनुक्रमांक के अतिरिक्त कुछ न लिखें।
5. प्रश्न पत्र कोड एवं प्रश्न पत्र ID सावधानी पूर्वक लिखें।
6. अपनी स्थिति स्पष्ट लिखें।
7. उत्तर पुस्तिका के पृष्ठों की संख्या देखें। अगर उत्तर पुस्तिका में पृष्ठ (1-24) से कम है या फटे हुए है, तो शुरू होने के पूर्व दूसरी उत्तर पुस्तिका ले लें।
8. प्रश्नपत्र को देख, यदि प्रश्नपत्र के विषय कोड, विषय का नाम तथा प्रश्न में कोई त्रुटि है तो उसके परी होने के 30 मिनट के अन्दर कक्ष निरीक्षक को तत्काल सूचित करें, उसके बाद विश्वविद्यालय द्वारा को नहीं की जायेगी।
9. प्रश्नों के उत्तर लिखने के लिये पेंसिल का प्रयोग न करें।
10. बी कोपी या अतिरिक्त ग्राफ नहीं दिया जायेगा।

INSTRUCTION TO THE CANDIDATE

1. Read the instructions carefully given on the Question Paper, Admit Card & Answer Script.
2. Do not write anything on back side of the cover page.
3. Write on both sides of pages of answer book.
4. Do not write anything on question paper except Roll Number.
5. Write Paper Code & Question Paper Id carefully.
6. CHECK the number of pages (1-24) or any other kind of damage in your answer script, if found than change the answer script immediately before th commencement of examination.
7. CHECK the Question Paper for any kind of discrepancy e.g. Subject Code, Name, and Question of the Question Paper during first THIRTY MINUTES commencement of the exam, so that it can be corrected in TIME. After that r corrections shall be entertained by the university.
8. Do not use pencil for answering the question.
9. Write status correctly e.g. those appearing in carry over papers should fill in s as Carry Over. Those appearing as Ex- Students should fill in status as ex.
10. No supplementary answer book & graph paper will be provided.

INSTRUCTION TO THE CANDIDATE FOR FILLING PART-IV

1. Use blue or black ball point pen for writing alphabets & numerals in ☐ Boxes.
2. Use blue or black ball point pen for filling the circles.

	1	8	1	5	4	3	2	1	6	9
0	0	0	0	0	0	0	0	0	0	0
1	●	1	●	1	1	1	1	●	1	1
2	2	2	2	2	2	2	●	2	2	2
3	3	3	3	3	3	●	3	3	3	3
4	4	4	4	4	●	4	4	4	4	4
5	5	5	5	●	5	5	5	5	5	5
6	6	6	6	6	6	6	6	6	●	6
7	7	7	7	7	7	7	7	7	7	7
8	8	●	8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9	9	9	●

Note- If your Roll No. is of 10 digits. Please leave first three columns .

Paper Code

1

SECTION-A

Answer-1

Structure of C Program

```
// Program for addition of two numbers // Documentation section
#include<stdio.h> // Pre-processor directive
#define MAX 100 // Definition section

int sum(int a, int b); // Global declaration section.

Main() //Main Method
{
 Int s, i, j;
 Printf("Enter two values");
 Scanf("%d %d", &i, &j);
 s= int sum(i, j);
}

int sum(int a, int b) // user defined function section
{
 int add;
 add = a + b;
 return add;
}
```

1. Documentation section

- This section tells us about the program. what is the program about and what does

it solve.

- It is used to make program more Interact to the user.
- It makes the program easily understandable by the user.

2. Preprocessor Directive Section

- The preprocessor directive section includes the pre processor directive (#) include <header file>
- It is used to add the predefined functions and libraries in the program.
- It is used to transform the program before the actual compilation.

3. Definition Section

- This function includes all the macros and their definitions.
- These include the name for the piece of code and as the name is encountered the compiler uses the piece of code of that macro.
- It also begins with (#)

4. Global Declaration Section

- This section includes the global declaration of the variables, functions, arrays, strings etc.
- The variables, functions etc that are required globally by the the program. is declared in this section.
- The variables and functions can be used anywhere in the program. in any function.

5. Main function.

- It is the function where the program flow enters and exits from the program.
- It includes the body enclosed within parenthesis ( ) curly pair of braces.
- It is used to perform specific tasks. It can include various function calls.
- It can include various variable, structure and function declarations.

6. User-defined Function

- This section is used to perform a specific task.
- It is a piece of code that can be used to perform a coherent task.
- It can be defined according to the user.

Answer - 2.

Variable

- Variables are the named memory locations.
- Variables are the piece of memory which is given name.
- These can be of any valid datatype.
- Which means that the variables are the named memory locations that are assigned with a data type.
- The variable store the value on a temporary basis

- Variables can be used as a substitute for the value they store in the program.
- Variables can be classified into various types based on their scope, datatype and functionality.

Syntax for variable declaration:

1. datatype variable_name;

ex. int i;
char c;
float f;

2. datatype variable1, variable2, variable3 ... ;

ex. int i, j, k;
char c, s, l;

Example:

```
#include<stdio.h>
main()
{
int n;  // variable declaration
printf("Enter a value");
scanf("%d", &n); // variable initilization.
printf("%d", n);
}
```

Rules for writing variable.

- A variable name shall not be identifier
- A variable can have the combination of alphabets (a-z) (A-Z), digits (0-9) & (_) underscore.
- It must not begin with a digit.
- It can must begin with an alphabet or digit followed by the combination of both & digits.
- Both the cases are treated differently in the variable declaration.
- No two variables of same name shall be declared in the same scope.
- Variables shall be of reasonable length.
- It must not contain any other special symbol except for (_) underscore.

Paper Code

7

Answer-C.

```
// C program for the biggest of three number
#Include<stdio.h>
  main()
  {
  int a, b, c;
printf("Enter three values to know which
        one is the biggest");
scanf("%d %d %d", &a, &b, &c);
if (a>b && a>c)
  {
    printf("the biggest digit is %d", a);
  }
else if (b>c && b>a)
  {
    printf(" the biggest digit is %d", b);
else
  {
    printf("the biggest digit is %d," c);
  }
}
```

However this code is sufficient · but if in case the three values are equal.
we can add an additional statement in the if-else-if ladder.

```
else if (a==b && a==c)
{
    printf("All the digits are equal.");
}
```

Answer - D.

Formatted input output statements.

- Formatted input output statements can be used to perform various input and output operation.
- We can use various formatted statement according to our needs.
- These are known as formatted statement because we can use the format specifier with these.
- various format specifiers include.
  %.d, %d, %.f, %.s.
- These statements can be used with

Paper Code

9

all types of data types
for example:- Integer, character, float, double.

- Various format specifiers statements include.

1. %printf():
2. scanf()
3. sprintf()
4. sscanf()

1). printf()

- It is a formatted & pad output statement.
- It can be used with all the valid data types.
- It is used to print the data on the console screen.
- We can print the data of our choice by using various format specifiers.

Example:

```
#include<stdio.h>
main()
{
int a=2;
int char='A';
printf("%d",a);
printf("%c", char);
```

output:
2
A

4

2). Scanf().

- It is a format ~~outp~~ input statement.
- It is used to take input from the keyboard from the user.
- It can take input of all the valid datatypes
- We use formatted specifiers for inputting data of required.

Example.

```
#include<stdio.h>
main()
{
int a,b;
~~scan~~ printf("Enter value");
Scanf("%d",&a); // taking 1st input
printf("Enter another value");
scanf("%d",&b); // taking second input
printf("%d%d",a,b);
}
```

3. Sprint().

- sprintf() is similar to printf but it prints the data in character array rather than the console.

- It stands for string printf()
- It can only be used with the character datatype

4. SScanf()

- It stands for string scanf()
- It can only be used with character datatype.
- It reads the data from the string or character array rather than console screen.

Example: sscanf("%c", c);

Answer:

```
// C Program to find the largest element in array.
#include <stdio.h>
main()
{
int i, j, n, l;
printf("Enter the length of array");
scanf("%d", &n);
int ar[n];

printf("Enter array elements\n");
for(i=0; i<n; i++);
{
  scanf("%d", &ar[i]);
}.
```

```
printf("the array is \n");
for(j=0;j<n;j++)
{
printf("%d  ",ar[i]);
}

l=ar[0];
for(i=1;i<n;i++)
{
 if(ar[i]>l)
 {
    l=ar[i];
 }
}
printf("the biggest elements is %d",l);
printf("it is found at the position %d",i+1);
}
```

Answer - 1.

function.

- A function is a piece of code that is used to perform a specific task.
- It is a self contained block of statement that can be used to perform a coherent task.

- The body of the function is written inside the matching pair of curly braces {}.

⇒ function ~~definition~~ declaration

syntax: datatype functionname(parameterlist);

* there can be zero or more parameters in the function declaration.

ex: int add (int, int);
int cal ( );

⇒ function ~~declaration~~ definition

Syntax: datatype function_name ( )
{
——;
——; // body of the function.
——;
}

ex: int add (int a, int b)
{
int c;
c=a+b;
}

⇒ function calling

while function calling the definition of the function called should be before the calling function.

Ex.

```
#include<stdio.h>
int add(int a, int b) ⟶ called function.
{
int s;
s=a+b;
return s;
}
main() ⟶ calling function.
{
int i=20;
int j=30;
printf("%d", add(i,j));
}
```

↳ function calling

User defined function.	Library function.
• These function are created by the user during programming.	These functions are the predefined functions.
• These functions are included by using #include "fun.name";	These function are included using #include<fun.name>,
• User defined function first need to declared & then used	Library functions are already declared.

• Modification according to user can be done	Modifications cannot be done.

Answer - 6.

- Pointer are the user defined datatype that are used to store the memory address of another variables, functions and even other pointer etc.
- Pointer is a variable that stores the location for another variable.
- It provides low level memory access & dynamic memory allocation.

1. pointer declaration.
   Syntax: * pointername;
   ex: * ptr;
   * i;

2. pointer initilization
   A pointer is initilized using a (&) operator. ampisand operator.
   Syntax: * pointer name = & variable name.
   ex: * ptr = & a.

```
#include <stdio.h>
main()
{
int a=10;
int *ptr; *ptr=&a;
```

Pointer derfrencing.

Pointer derfrencing is the proces of returing the value of the address location that the pointer holds.

ex. #include<stdio.h>.
main()
{
int a=10
int *ptr=&a;
printf("%d", *ptr);
}

output
10.

↓
pointer derfrencing.

	Global Variables	Local variables.
1.	These are declared outside any function	They are declared inside the functions
2.	This life remains throughout the program	Their life begins from their declaration & ends with the end of the function
3.	They are stored in data segment of memory	They are stored in stack memory
4.	No global variable of same name can be used in program	same name variable can be used in program but in different scope

8

5. Any changes made in global variable is reflected throughout the program.	The changes are only reflected in the scope.
6. They are initialized with zero.	They are initialized with garbage value.
7. Their life/scope is throughout the program.	Their life/scope is restricted to the function only.
8. They can be accessed in all functions.	They cannot be accessed outside function.

Answer-I

Structures.

Structure are the user defined datatypes. which can combine different datatypes under a same time.

1. We used structures because we can combine different type of data irrespective of their datatype into the same type.

2. Structure helps us in organizing the data irrespective of the datatype.

3. Structure can be used to form custom datatypes like complex numbers- datas, time etc.
4. Structure is an easy approach to the programs.

the general syntax for different datatypes under same datatype is :

```
struct structurename
{
  int a;
  char b;
  float c;
  char z[100];
} v1, v2, v3;
```

} different datatypes under same type struct.

These member can be accessed by each variable by (.) dot operator.

$v_1.a = 2$, $v_1.b = A$. etc.

Section-B

Operators are the special symbols that are used to perform various operations on operands.

1. unary operator : &, &&, ||, ++, --
2. Binary operators : +, -, /, %, *
3. Ternary operator : ?:

Types of operator.

1. Arithmatic operator
2. Logical operator.

3. Relational operator.
4. Bitwise operator.
5. Shift operator.
6. Comma Operator.
7. Miss operator
8. Assignment operator
9. Conditional operator

1. Arithmatic operator.
These are the unary operators that are used to perform arithmatic tasks on the operands. like addition (+) substraction (-), multiplication (*), division (/), (%) mod operation, (++) increment, (--) decrement

Increment and decrement have two types.
1- pre increment and pre decrement (++2, --2)
2- post increment & post decrement. (2++, 2--)

2. Logical Operators
These operators are used to combine two operands logically. There are three logical operator.

Logical OR (||)
This returns true if any of operand is true

A	B	A || B
0	0	0
0	1	1
1	0	1

Logical AND (&&)
this returns true only if both operand

A	B	A && B
0	0	0
0	1	0
1	0	0

Logical Not (!)
This negates the value of operand.

A	!A
0	1
1	0

3. Relational operator
This operator is used to find the relation between any two operands.
There are various relational operand.

operator		if A=3 & B=5
>	greater than	A>B = 0
<	less than	A<B = 1
>=	Greater than equal to	A>=B = 0
<=	less than equal to	A<=B = 1
==	equal to	A==B = 0
!=	not equal to	A!=B = 1

4. Assignment operator
This operator is used to assign values to the operands.

operator	functionality	
=	A=B	A=B
+=	A+=B	A=A+B
-=	A-=B	A=A-B
%=	A%=B	A=A%B
*=	A*=B	A=A*B
/=	A/=B	A=A/B.

5. Bitwise operator.
This operator is used to perform operations of bits there are following operator.

Bitwise AND (&)
Returns 1 if both bits are 1

0	0	0
0	1	0
1	0	0
1	1	1

Bitwise OR (|)
Returns 1 if any bits is 1

0	0	0
0	1	1
1	0	1
1	1	1

Bitwise not (~)
Returns the rejection of bit.

0	1
1	0

Bitwise (^) xor
If both bits are opposite it returns 1

0	0	0
0	1	1
1	0	1
1	1	0

6 Shift operator.

This operator shifts the values written on the left by the value as written at the right.

Left shift operator

Syn. (a<<b)

(2<<1)

0010 <<1

0100.

Right shift operator.

Syn. (a>>b)

(2>>1)

0010 >>1

0001

7. Comma operator.

This operator works in a way that it evaluates the first operands negates it & then evaluates the second operand.

```
a=6, b=7;
printf("%d", (a,b));
```

output = 7.

8 Conditional operator.

This operator is a ternary operator if the test expression results to true it evaluates the first operand & if the condition evaluates to false then third operand is executed.

Syntax - ?:

(test expression)? Value1 : value2;

Section 2

Answer - 7.

- String is a ~~char~~ array of character datatype which is followed by a '\0' null character.
- A string can contain zero or more characters but it always ends with a '\0' null character.

1. ~~The~~ Declaration of String.

Syntax: datatype string name [size];

char s[6];

char s[10];

2. Initilization of String.

1). Initilization with declaring size.

char str[6] = "Hello";

2). Initilization without declaring size.

char str[] = "Hello";

Here the compiler considers the size by analyzing the number of elements in the array.

3). character by character initilization with size.

char str[6] = {'H', 'E', 'L', 'L', 'O', '\0'};

4). character by character initilization without size.

char str[] = {'H', 'E', 'L', 'L', 'O', '\0'};

String manipulation functions.

1). strlen()

This function is used to calculate the length of the string

Its return type is integer.

Its argument type is string.

syntax: strlen(s1);

eg: #include <stdio.h>

{

main()

char s1[:] = "Helloworld";

int len;

len = strlen(s1);

printf ("the length of string is %d", &len);

Output: The length of string is 11

2. Strlwr()

- This function is used to convert the characters of the string into lowercase.
- If the characters are already in lowercase, then they are left unaffected.
- Its argument type is character.
- Its return type is also character.

ex: #include <stdio.h>

```
main()
{
char S1[7] "programming";
char S2[] = "CAPITAL";
char S3[7] = "";
S3 = strlwr(S1)
printf ("%s", S3);
S3 = strlwr(S2);
printf (strlwr(S3);
}
```

Output: programming capital.

3) StrUpr

- This function is used to convert all the characters of the string into Upper case.
- If the characters are already in uppercase then it does not affect them.
- Its return type is String
  Its argument type is also String.

#include <stdio.h>

```
main()
{
char S1[] = "programming";
char S2[] = strlwr(S1);
printf ("%s", S2);
}
```

Output: PROGRAMMING.

4. Strcpy()
- this function is used to copy the string of the second argument into first argument string.
- It has two arguments.
- It's return type is also a string.

Syntax: strcpy(s1,s2);

```
#include<stdio.h>
main()
{
  char s[]="Hello";
  char s2[]="";
strcpy(s,s2);
printf("%s",s);
}
```

Output: Hello.

5. Strcat()
- This function is used to concatenate two strings.
- It joins the two string & stores it into the first string and returns the first string.

Syntax: strcat(s1,s2).

6. Strstr(s1,s2)

It is used to find the first occurence of s1 in s2.