

SECOND SEMESTER EXAMINATION 2010-11**COMPUTER CONCEPTS & PROGRAMMING IN C****Time: 3 Hours****Total Marks: 100**

Note: (1) *This question paper consists of Three sections. Section-A contains objective type questions and carries 20 marks. Section-B consists of short answer type questions which are of 30 marks and Section-C contains long answer type questions of total of 50 marks.*

(2) *Your answers for Sections B and C should be precise and to the point.*

(3) *Answer to the questions each section must be done at one place in your answer book.*

(4) *You are required to attempt all the questions.*

SECTION—A

1. There are a total of 10 multiple choice questions. Only one of the answers out of given four choices is correct. Write the correct answer: (10 × 1 = 10)

(i) The example of system software is:

- (a) UNIX (b) Compiler
(c) MS- DOS (d) All of the above.

Ans. (d)

(ii) Devices used for producing hard copy is:

- (a) Printers only
(b) Plotters only
(c) Scanners only
(d) Printer and Plotters both.

Ans. (a)

(iii) Which of the following is an example of scripting language:

- (a) LISP (b) C++
(c) PERL (d) COBOL

Ans. (a)

(iv) How many bytes are occupied by the string literal constant "abc" in memory:

- (a) 1 byte (b) 3 bytes
(c) 4 bytes (d) 2 bytes

Ans. (c)

(v) Which of the C construct is used to terminate a loop in the middle:

- (a) return statement
(b) continue statement
(c) break statement
(d) All of the above.

Ans. (c)

(vi) Minimum number of times a do while loop will execute:

- (a) 2 times (b) 1 time
(c) 0 time (d) None of the above.

Ans. (b)

(vii) By default, array index in C language starts from;

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

Ans. (c)

(viii) The complement of the relational operator (a == b) is:

- (a) (a <= b) (b) (a >= b)
(c) (a != b) (d) All of the above.

Ans. (c)

(ix) By default the return type of a function is:

- (a) char (b) float
(c) int (d) void.

Ans. (c)

(x) Which of the following is an example of secondary memory:

- (a) RAM
- (b) ROM
- (c) Cache Memory
- (d) None of the above.

Ans. (d)

2. State whether the following statements are TRUE or FALSE: (5 × 1 = 5)

(i) Floppy disk is an example of main memory.

Ans. False

(ii) The array is used to store the elements of similar data type.

Ans. True

(iii) An entry controlled loop is executed at least once.

Ans. False

(iv) Type of a function depends upon its arguments type.

Ans. False

(v) ALU is integral component of CPU.

Ans. True

3. Fill in the blanks: (5 × 1 = 5)

(i) The scanner is an example of device.

Ans. Input

(ii) PROM is an example ofmemory.

Ans. Main

(iii) An integer pointer variable is declared as

Ans. integer

(iv) The < < is an example of operator.

Ans. left shift

(v) The octal equivalent of $(100)_{10}$ is

Ans. $(144)_8$

SECTION—B

4. There are Seven questions in this section. Attempt any Five questions: (6 × 5 = 30)

(a) What are the different types of operators in C language?

Explain with example. Discuss the significance of each.

Ans. The different types of C language operators are

- ASSIGNMENT OPERATOR
- BITWISE OPERATORS
- INCREMENT AND DECREMENT OPERATORS
- RELATIONAL OPERATORS
- LOGICAL OPERATORS

ASSIGNMENT OPERATORS: The assignment operator is used to assign the value of a variable or expression to a variable. The syntax of assignment operators is

var = expression;

var = var;

BITWISE OPERATORS: The Bitwise operator interprets operands as strings of bits. Bit operations are performed on this data to get the bit strings. These bit strings are then interpreted according to data type. There are six bit operators, which are defined as follows:

Bitwise AND(&)

Bitwise OR(|)

Bitwise XOR(^)

Bitwise complement(~)

Left shift(<<)

Right shift(>>)

A demonstration of C bitwise operators

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
void main()
```

```
{
```

```
int dl = 4;
```

```

intd2 = 6,/* 110*/
intd3;
printf("\nd1=%d",d1);
printf("\nd2=%d",d2);
d3 = d1 & d2;
printf("\n Bitwise AND d1 & d2 = %d",d3);
d3 = d1 | d2; printf("\n Bitwise OR d1 | d2 = %d",
d3);
d3 = d1 ^ d2; printf("\n Bitwise XOR d1 ^ d2 =
%d", d3);
d3 = ~d1;
printf("\n Ones complement of d1 = %d", d3);
d3 = d1 << 2; printf("\n Left shift by 2 bits d1 << 2
= %d", d3);
d3 = d1 >> 2; printf("\n Right shift by 2 bits d1 >>
2 = %d",d3);
}

```

INCREMENT AND DECREMENT OPERATORS

The increment and decrement operators are used to increase and decrease the value of variable. The increment and decrement operators support in both prefix and postfix form.

Here is syntax of increment and decrement operators:

Let us consider a variable count then.

The syntax for increment operators is

```
++count;
```

```
count++
```

and the syntax for decrement operators is

```
-- count;
```

```
count--
```

RELATIONAL OPERATORS

The relational operator is allowed to the comparison of two or more variables. The relational operators are

```
== equalto
```

```
!= notequal
```

```
< lessthan
```

```
<= lessthan orequalto
```

```
> greaterthan
```

```
>= greaterthan orequalto
```

LOGICAL OPERATORS

The logical operator is used to use the multiple relational operators or logical operators into one boolean result. Programming language C supports the negation (!), logical AND (&&), and logical OR (||). The order of precedence is !, &&, ||. Here is the logic table of logical operators:

A program of assignment operator

```

#include <stdio.h>
void main(){
int x=10;
int y = x; printf("y = %d\n",y);
y += 10; printf("y += 10;y = %d\n",y);
y -= 5; printf("y -=5;y = %d\n",y);
y *=4;
printf("y *=4;y = %d\n",y); y /=2; printf("y /=2;y
= %d\n",y);
}

```

(b) Write an algorithm to print all the even numbers and odd numbers between any given two integers N_1 and N_2 ($N_1 < N_2$) and also print the sum of all even numbers and odd numbers.

Ans. begin

read N_1, N_2

sum = 0

for $i = N_1$ to N_2

if ($N_i \% 2 = 0$)

begin

sum = sum + i

print i

end

for $i = N_1$ to N_2

if ($i \% 2 \neq 0$)

begin sum = sum + i

print i

```

end
print sum
end

```

(c) Write short note on the following with example in reference to C language:

(i) Data types

Ans. Please See Q. 2(b) of first semester 2008-09.

(ii) Entry and exit control loops,

Ans. The entry control statements are used in loop statements to show the entry point to execute the looping statements and exit control statements are used to exit the statements.

For example

```

sum = 0;
a = 1;
while (a <= 10)
{
    sum = sum + a;
}

```

Here if the value of a is less than 10 then loop will execute otherwise loop will terminate.

(iii) Switch statement and if statement,

Ans. The if Statement

The if conditional statement is used to provide the select ional control structure that executes a set of statements based on a condition. The syntax for if statements is

```

if(condition)
{
    set of statements;
}
next statement;

```

Example:

```

if(marks < 30)
{
    printf(" You fail\n");
}

```

The switch Statement

The switch statement is another form of the multi way decision statement. It is well structured, but can only be used in certain cases where; It is very similar to if statement but can not replace in all cases.

The syntax for switch case is

```

switch( integer variable or character variable or
integer expression)

```

```

{
    case integer or character constant 1 : set of
statements;
    break;
    ... ..
    case integer or character constant2 : set of
statements;
    break;
    case integer or character constant-n : set of
statements;
    break;
    default: set of statements;
    break;
}

```

(d) (i) Write an algorithm to print the first 100 Fibonacci numbers and their sum.

Ans.

```

begin sum = 1
a = 0
b = 1
print a
print b
c = a + b
for i = 1 to 98
begin
print c
a = b
b = c
sum = sum + c

```

```

c = a + b
end
print c
end

```

(ii) Discuss various storage classes in C with suitable example. Also give their significance.

Ans. There are four types of storage classes in C. It defines the scope and lifetime of a variable or function.

1. **auto:** This is the default storage class. Auto can only be used with in functions, i.e. only for local variables, not for

globals.

Example

```
auto int x;
```

2. **register:** The variables declared using the register storage class may be stored in cpu registers instead of RAM. Since it doesn't have a memory location, the '&' operator for getting the address of the variable cannot be applied (in C). This storage class cannot be used for global scope data.

Example register int y;

3. **static:** This is the default storage class for global variables. In case of local variable, it is initialized at compile time and retains its value between the calls. By default the static variables will be initialized to zero, incase of pointer variable initialized to NULL.

Example static int z;

4. **extern:** Defines the global variable that is visible to all object modules. This type of variable cannot be initialized, since it is pointing to a storage location, where it is previously defined.

Example extern int u;

(e) Draw the flow chart and write a function in C to calculate the sum S of all the following series:

$$S = 1^1 + 2^2 + 3^3 + 4^4 + \dots + N^N$$

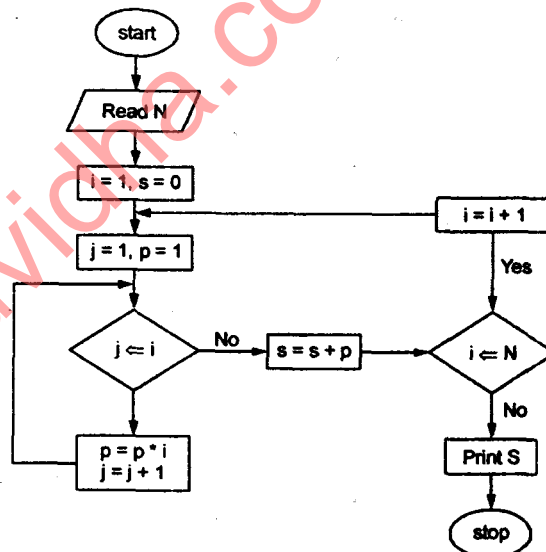
(N is a positive integer)

Ans. A function in C to calculate the sum of a given series is shown below

```

series_sum (1, N)
{
    int i, sum = 0, p;
    for (i=1; i <=N; i++)
    {
        p = pow(i, i)
        sum = sum + p
    }
    printf("%d", sum);
}

```



(f) Discuss the major components of a digital computer with suitable block diagram. Also discuss the functions of these components.

Ans. Please See Q. 2(b) (ii) of First Semester 2009-10.

(g) Give the flow chart and algorithm to calculate the number of words in a given sentence.

Ans. Algorithm to count the number of words

begin

len=0

count=0;

```

read str
compute string length;
for i=1 to length
    begin
        if(str[i]!="")
            count++;
    end
count++;
print count
end

```

5. This section contains seven parts. Attempt any five parts: (10 × 5 = 50)

- (a) Write a program in C to store the floating point numbers in two matrixes A and B of size 4×4 each. Further the program should compute the summation and multiplication of the two matrixes and store the summation and multiplication in matrix C and D respectively.

Ans.

```

#include<stdio.h>
#include<conio.h>
main()
{
    int a[3][3],b[3][3],i,j,k;
    static int c[3][3],d[3][3];
    clrscr();
    printf(" Enter the first matrix elements :\n");
    for(i=0;i<3;i++)
        for(j=0;j<3;j++)
            scanf("%d",&a[i][j]);
    printf("\nEnter the second array elements :\n");
    for(i=0;i<3;i++)
        for(j=0;j<3;j++)
            scanf("%d",&b[i][j]);
    for(i=0;i<3;i++)
        for(j=0;j<3;j++)

```

```

{
    c[i][j]=a[i][j]+b[i][j];
    for(k=0;k<3;k++)

        D[i][j]=d[i][j] + a[i][k]* b[k][j];
    }
    printf("\nThe addition of two array elements is :\n");
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
            printf("%4d",c[i][j]);
        printf("\n");
    }
    printf("\nThe multiplication of two array elements is :\n");
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
            printf("%4d",d[i][j]);
        printf("\n");
    }
    getch( );
}

```

- (b) What do you mean by sorting ? Write a program in C to sort the given n positive integers. Also give the flow chart for the same.

Ans. Sorting is the process of putting a list or a group of items in a specific order. Some common sorting criteria are: alphabetical or numerical. Sorting can also be done in ascending order (A-Z) or descending order (Z-A).

```

#include<stdio.h>
void main()
{
    int A[20],N, Temp, i,j;
    clrscr();
    printf("\n\n\t ENTER THE NUMBER OF TERMS...:");
    scanf("%d" &N);

```

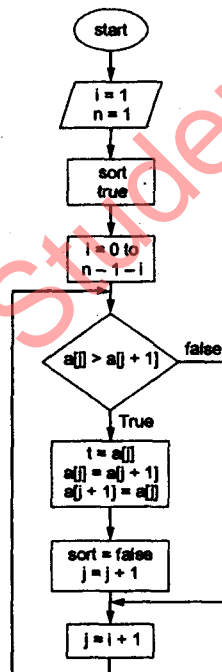
```
printf("\n\t ENTER THE ELEMENTS OF THE  
ARRAY....");
```

```
for(i=0; i<N; i++)  
{  
gotoxy(25, 11+i);  
scanf("\n\t\t%d", &A[i]);  
}  
for(i=0; i<N-1; i++)  
for(j=0; j<N-i; j++)  
if(A[j]>A[j+1])  
{  
Temp = A[j];  
A[j] = A[j+1];  
A[j+1] = Temp;  
}
```

```
printf("\n\tTHE ASCENDING ORDER LIST  
IS....\n");
```

```
for(i=0; i<=N; i++)  
printf("\n\t\t\t%d", A[i]);  
getch();  
}
```

A flow chart of bubble sort



(c) (i) Write a program in C that reads the two strings of length at least 7, then concatenate these strings.

Ans.

```
#include<stdio.h>  
#include <string.h>  
main()  
{  
char str1 [20], str2[20], i=0, j=0;  
clrscr();  
printf("Enter the first string :\n");  
gets(str1);  
printf("Enter the second string :\n");  
gets(str2);  
while(str1[i]!='\0')  
i++;  
while(str2[j]!='\0')  
{  
str1[i]=str2[j];  
j++;  
i++;  
}  
str1[i]='\0';  
printf("\n The concatenating string is :\t%d",  
puts(str1);  
getch();  
}
```

(ii) Write a program in C that takes a year from twentieth century as an input and then tells whether it is a leap year or not?

Ans.

```
#include <stdio.h>  
main()  
{  
int year;  
printf("Enter a value year:\t");  
scanf("%d", &year);
```

```

    if(((year % 4==0) &&( year % 100 !=0)) || (year
% 400==0 ))
    printf("It is a leap year");
    else
    printf("It is not a leap year");
    return 0;
}

```

- (d) (i) What do you mean by pointers ? How pointer variables are initialized? Write a program in C to swap the values of any two integer variables using pointers.

Ans. Pointers are widely used in programming; they are used to refer to memory location of another variable without using variable identifier itself. They are mainly used in linked lists and called by reference functions.

Declaring pointers can be very confusing and difficult at times. To declare pointer variable we need to use * operator (indirection/dereferencing operator) before the variable identifier and after data type. Pointer can only point to variable of the same data type.

The pointer variables are initialized by

```
Int *temp = NULL
```

A program to swap two variables

```

#include<stdio.h>
#include<conio.h>
void main()
{
    int a=2,b=3;
    swap(a,b);
    printf("%d%d",a,b);
    getch()
}
swap(int *x,int *y)
{
    int t;
    t=*x;
    *x=*y;

```

```

*y=t;
printf("%d%d",x,y);
}

```

- (ii) Write a function in C that finds the reverse of a given integer number.

Ans. A PROGRAM TO FIND THE REVERSE OF A GIVEN NUMBER

```

#include
#include
void main()
{
    clrscr();
    int r=0,d,m,n;
    printf("Enter a value:");
    scanf("%d",&n);
    m=n;
    do
    {
        d=m%10;
        m= m/10;
        r=(r*10)+d;
    }
    while(m!=0);
    printf("%d is the reverse \n",r);
}
getch();
}

```

- (e) Create the database of student in C having the following attributes: Roll_no_name, Stud_address, Stud_city, Stud_PINCode, Stud_sem, Rank, and Branch. Also write the program in C to enter the data for 500 students in any order and then display the list of students for a given branch and semester on display.

Ans.

```

#include<stdio.h>
#include<conio.h>
#define MAX 2

```



```

struct student
{
    int roll_no;
    char stud_name[20];
    char stud_address[20];
    char stud_city[10];
    int stud_pincode;
    int stud_sem;
    int rank;
    char branch[10];
}

main()
{
    struct student s[500];
    int i,j;
    int troll_no;
    char tstud_name[20];
    char tstud_address[20];
    char tstud_city[10];
    int tstud_pincode;
    int tstud_sem;
    int trank;
    char tbranch[10];
    clrscr();
    printf("\nPlease enter the details of students \n");
    for (i=0;i<1;i++)
    {
        printf("\nEnter Roll No.:");
        scanf("%d",&s[i].roll_no);
        printf("\nEnter Name and Address :");
        scanf("%s%s",s[i].stud_name,s[i].stud_address);
        printf("\nEnter City and Pincode :");
        scanf("%s",s[i].stud_city,s[i].stud_pincode);
        printf("\nEnter semester and Rank:");
        scanf("%d%d",&s[i].stud_sem,&s[i].rank);
        printf("\nEnter Branch:");
        scanf("%s",s[i].branch);
    }

```

```

    for(i=0;i<5;j++)
    {
        if(s[j].stud_sem>s[j+1].stud_sem)
        {
            tstud_sem=s[j+1].stud_sem;
            s[j+1].stud_sem=s[j].stud_sem;
            s[i].stud_sem=tstud_sem;

            troll_no=s[j+1].roll_no;
            s[j+1].roll_no=s[j].roll_no;
            s[j].roll_no=troll_no;

            trank=s[j+1].rank;
            s[j+1].rank=s[j].rank;
            s[j].rank=trank;

            strcpy(tbranch,s[j+1].branch);
            strcpy(s[j+1].branch,s[j].branch);
            strcpy(s[j].branch,tbranch);

            strcpy(tstud_name,s[j+1].stud_name);
            strcpy(s[j+1].stud_name,s[j].stud_name);
            strcpy(s[j].stud_name,tstud_name);

            strcpy(tstud_address,s[j+1].stud_address);
            strcpy(s[j+1].stud_address,s[j].stud_address);
            strcpy(s[j].stud_address,tstud_address);

            strcpy(tstud_city,s[j+1].stud_city);
            strcpy(s[j+1].stud_city,s[j].stud_city);
            strcpy(s[j].stud_city,tstud_city);

            strcpy(tstud_pincode,s[j+1].stud_pincode);
            strcpy(s[j+1].stud_pincode,s[j].stud_pincode);
            strcpy(s[j].stud_pincode,tstud_pincode);

```

```

    }
}
for(i=0;i<5;i++)
for(j=0;j<5-(i+1);j++)
{
    if(strcmp(s[j].branch,s[j+1].branch>0) &&
(s[j].stud_sem==s[j+1].stud_sem))
    {
        strcpy(tbranch,s[j+1].branch);
        strcpy(s[j+1].branch,s[j].branch);
        strcpy(s[j].branch,tbranch);

        tstud_sem=s[j+1].stud_sem;
        s[j+1].stud_sem=s[j].stud_sem;
        s[j].stud_sem=tstud_sem;

        troll_no=s[j+1].roll_no;
        s[j+1].roll_no=s[j].roll_no;
        s[j].roll_no=troll_no;

        trank=s[j+1].rank;
        s[j+1].rank=s[j].rank;
        s[j].rank=trank;

        strcpy(tstud_name,s[j+1].stud_name);
        strcpy(s[j+1].stud_name,s[j].stud_name);
        strcpy(s[j].stud_name,tstud_name);

        strcpy(tstud_address,s[j+1].stud_address);
        strcpy(s[j+1].stud_address,s[j].stud_address);
        strcpy(s[j].stud_address,tstud_address);

        strcpy(tstud_city,s[j+1].stud_city);
        strcpy(s[j+1].stud_city,s[j].stud_city);
        strcpy(s[j].stud_city,tstud_city);

```

```

        strcpy(tstud_pincode,s[j+1].stud_pincode);
        strcpy(s[j+1].stud_pincode,s[j].stud_pincode);
        strcpy(s[j].stud_pincode,tstud_pincode);
    }
}
for(i=0;i<5;i++)
{
    printf("%d\t",s[i].roll_no);
    printf("%s\t%s",s[i].stud_name,s[i].stud_address);
    printf("%s\t%s",s[i].stud_city,s[i].stud_pincode);
    printf("%d\t%d\t",s[i].stud_sem,s[i].rank);
    printf("%s\n",s[i].branch);
}
getch();
}

```

**(f) (i) What do you mean by parameter passing?
Discuss various types of parameter passing
mechanism in C with example.**

Ans. When a function is called and if the function accepts some parameters, then there are two ways to receive the parameters:

Pass by value or call by value

Pass by reference or call by reference

Pass by value or call by value

In this you can use the concept of actual and formal parameter. It means the value of actual parameter is passed and they are copied into the formal parameters. Since the value of actual and formal parameter is stored in different memory locations so the changes in formal parameter do not change the value of actual parameters. For example, Example II: Simple program of formal and actual argument

```

#include<stdio.h>
#include<conio.h>
int compute( int, int );
main()
{

```

```

int number1 = 100, number2 = 500;
printf("The value before calling the function");
printf(" number1 = %d and number2 = %d \n",
number1 , number2);
compute(number1, number2 );
printf("The value after calling the function");
printf("number1 = %d and number2 = %d \n",
number1 , number2);
}
int compute( int num1 , int num2 )
{
    num1= num1 + 100;
    num2=num2+ 100;
}

```

Pass by reference or call by reference

In this concept instead of sending the values of actual parameter you send the address of the actual parameter to the functions. The changes that are made to formal parameters also change the value of actual parameters. For example

```

#include <stdio.h>
#include <conio.h>
int main()
{
    int a;
    int b;
    printf("Enter the value of a and b:");
    printf("%d%d",&a,&b);
    swap(&a,&b)
    printf("The value of a and b after swap is %d
and %d', a, b);
    getch();
}
void swap(int*px,int*py)
{
    int temp;
    temp = *px;
    *px = *py;

```

```

*py = temp;
}

```

- (ii) What do you mean by recursion? Write a recursive function to calculate the factorial of a given integer.

Ans. A function which calls itself is called recursive function. This function is used to solve certain problems which are recursive in nature.

```

#include<stdio.h>
long factorial(long);
void main()
{
    long int number;
    printf("\nEnter an integer value;");
    scanf("%ld",&number);
    printf("\nThe factorial of %ld is %ld\n", number,
factorial (number));
}
long factorial (long x)
{
    if (x ==1)
        return 1;
    else
        return x*factorial(x-1);
}

```

- (g) (i) Write a program that counts the total number of vowels in a given sentence.

Ans.

```

#include
#include
void main()
{
    int a=0, e=0, i=0, o=0, u=0,sum=0;
    char c;
    clrscr();
    printf("Enter string;-");
    printf("\nString will be terminated if you press
Ctrl-Z\

```

