Ajay Kumar Garg Engineering College, Ghaziabad

Department of CSE

Model Solution- ODD Semester (2017-18)

Sessional Test -2

Subject Code	:	RCS-101
Subject Name	:	Computer System and Programming in C
Names of Faculty Teaching with Signature		Mr. Rajeev Singh Mr. Deepak Rai

Name and Signature of Hod:

Mamta Bhusry

Prof. Mamta Bhusry

Section – A

Q.1) Differentiate between break and continue statement.

Ans:-

Break	Continue
• A break can appear in both switch and loop.	• A continue can appear only in loop.
• A break causes the loop or switch to terminate immediately it is encountered.	• A continue doesn't terminate. It causes the loop to go to the next iteration, it just skip the statements in the loop that appear after it.

Ex:- `for(i=1; i<=10; i++)`
`{ if(i==4)`
`break`
`printf("%d", i);`
`}`
Output: 1 2 3
	Ex:- `for(i=1; i<=10; i++)`
	`{ if(i==4)`
	`continue;`
	`printf("%d", i);`
	`}`
	Output: 1 2 3 5 6 7 8 9 10

Q.2) Differentiate between while and do-while loop.

Ans:-

while	do-while
• It is entry-controlled loop.	• It is exit-controlled loop.
• Condition is checked first.	• Condition is checked later.

• Ex:- `i=1;`
`while (i>5)`
`{ printf("%d", i);`
`i++;`
`}`
	• Ex:- `i=1;`
	`do`
	`{ printf("%d", i);`
	`i++;`
	`}while(i>5);`

Output: Nothing will print	Output: 1 will get printed even if condition is false.

Q.3) What is operator precedence and operator associativity.

Ans:- If more than one operators are involved in an expression, C has predefined rule of priority for the operators. This rule of priority is called as operator precedence.

If two operators of same precedence (priority) is present in an expression, then Associativity of operators indicate the order in which they execute. It is either Left to Right or Right to Left.

Q.4) WAP to print all the multiple of 7 between 1 to 100.

Ans:-

```
void main()
{
  int i;
  for(i=1; i<=100; i++)
  {
    if(i%7==0)
    {
      printf("%d\n", i);
    }
  }
}
```

Q.5) Find the output of the following program:

```
int a;
for(a=1; a<=5; a++);
{
  printf("%d", a);
}
```

Ans:- Output will be → 6

Section-B

Q.6) Explain different fundamental data types in terms of size, range and format specifiers.

Ans:-

Data type	Size (in bytes)	Range	Format Specifier
signed char	1	-128 to 127	%c
unsigned char	1	0 to 255	%c
signed short int	2	-32768 to 32767	%hd
unsigned short int	2	0 to 65535	%hu
int	2	-32768 to 32767	%d
long int	4	$-(2^{31})$ to (2^{31})	%ld
float	4	3.4E-38 to 3.4E+38	%f
double	8	1.7E-308 to 1.7E+308	%lf
long double	10	3.4E-4932 to 1.1E+4932	%Lf

Q.7) Define recursive function. WAP to print the factorial of a given number using recursion.

Ans:- Recursion: A recursive function is defined as a function that calls itself to solve a smaller version of its task until a final call is made which does not require any further call to itself.

Programs:

```
void main ( )
{
  int num;
  long int fact = 0;
  long int factorial (int);
  printf ("Enter the number = ");
  scanf ("%d", &num);
  fact = factorial (num);
  printf ("\n Factorial of %d is = %ldd", num, fact);
}

long int factorial (int n)
{ if (n==1)
    return 1;
  return (n * factorial (n-1));
}
```

Q.8) What is storage class? Explain different types of storage classes.

Ans :- Storage class defines the following things:

- scope i.e which all functions, the value of the variable would be available.
- default initial value, i.e if we do not initialize the variable, then what will be its default value.
- lifetime of that variable, i.e how long will that variable exist.

There are four storage classes:

* Automatic variables.
* External variables
* Static variables.
* Register variables.

* Automatic

- Scope: local to the method / block in which it is defined.
- Default Value: Garbage
- Lifetime: Till the end of the block in which it is defined.

* External:

- Scope: Global i.e everywhere in the program.
- Default Value: 0 (zero)
- Lifetime: Till the whole program finishes.

- Extern keyword is used before a variable to inform the compiler that this variable is declared somewhere else. The extern declaration does not allocate storage for variables.

* Static
  - Scope: local to the block in which it is defined.
  - Default value: 0 (zero)
  - Life time: Till the end of whole program.

Static variable tells the compiler to persist the variable till the end of program. Instead of creating and destroying a variable every time when it comes into and goes out of scope, static is initialized only once remains till end.

* Register:
  - Scope: local to the function in which defined.
  - Default value: garbage.
  - Life time: till the end of function where it is defined.

Q.9) What is a function? Explain different ways of passing values to a function with example.

Ans:- A large C program is divided into basic building blocks called functions. Functions contains set of instructions enclosed by "{ }" which performs specific operations in C program.

There are two ways of passing values to a C function:

- Call by value
- Call by reference.

✓ Call by Value:

- In call by value the value of the variable is passed to the function as parameter.
- The value of the actual parameter cannot be modified by formal parameter.

Ex:-

```
void main()
{
  int a = 10;
  void fun(int);
  printf("Value of a before call = %d", a);
  fun(a);
  printf("Value of a after call = %d", a);
}
void fun(int b)
{
  b = b * 10;
}
```

Output:

Value of a before call = 10
Value of a after call = 10

} no change

Call by reference

- In this, the address of the variable is passed to the function as parameter.
- The value of actual parameter can be modified by formal parameter.

Ex:-

```
void main()
{
  int a=10;
  void fun(int *);
  printf("Value of a before call=%d", a);
  fun(&a);
  printf("Value of a after call=%d", a);
}

void fun(int *b)
{
  *b = *b * 10;
}
```

Output:

Value of a before call = 10
value of a after call = 100

} change

Q.10) WAP to calculate the sum of following series:

$$x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \frac{x^5}{5} - \frac{x^6}{6} + \cdots + \frac{x^n}{n}$$

Ans:-

```
void main()
{
int x, esum=0, osum=0, tsum, i;
printf("Enter the value of x=");
scanf("%d", &x);
for(i=1; i<=n; i++)
{
    if (i%2==0)
      esum = esum - ((pow(x,i))/i);
    else
      osum = osum + ((pow(x,i))/i);
}
tsum = esum + osum;
printf("Sum of series = %d", tsum);
}
```

Section-C

Q. 11) Explain different types of operators in detail with suitable example.

Ans:- An operator is a symbol that tells the compiler to perform certain mathematical or logical manipulations.

Different operators are:-

* Arithmetic Operators

+, −, *, /, %

int a = 72, b = 9;

a + b → 81

a − b → 63

a * b → 648

a / b → 8

a % b → 0

* Unary Operators

++ (increment), -- (decrement)

post → use then change

pre → change then use

a++ means first use a then change a = a+1

++a means first change a = a+1 then use a.

Relational Operators

$<$, $>$, $<=$, $>=$

Returns 1 for true and 0 for false.

int a=5 , b=3

a < b → 0
a > b → 1
a <= b → 0
a >= b → 1

Equality: == , !=

int a=5, b=3;

a==b → 0
a!=b → 1

Logical Operators:

&& (logical AND)
|| (logical OR)
! (logical NOT)

int a=5, b=3, c=6

(a>b) && (b<c) → 1
(a>c) || (b>c) → 0
!a → 0

Bitwise operators:

& (bitwise AND)
| (bitwise OR)
~ (bitwise NOT)
<< (shift left)
>> (shift right)
^ (bitwise XOR)

✓ Assignment Operators: =, +=, -=, *=, /=, %=, &=, |= etc.

✓ Conditional Operators: ? : → false part
→ true part

Like: int a=5, b=3, c=0;

c = (a>b) ? 1 : 0;

Output: c=1

✓ size of operator: returns the size

Ex:- int a;
sizeof(a) → 2;

✓ comma operator: It evaluates first expression then ignores and then evaluate second expression and returns as result.

a = 5, 6, 7; → finally a will be 7.

Q.12) (a) WAP to convert binary number to decimal number using function.

Ex:- Binary number:- 110110

↓

decimal → 54

```
void main ( )
{
   void bin2dec ( void);
   bin2dec ();
}

void bin2dec (void)
{
  int n, m, num=0, p=1, r
  printf("\n Enter the binary number =");
  scanf("%d", &n);
   m=n;
   while (m>0)
    {
      r= m%10;
      num= num + (p*r);
       p= p*2;
       m= m/10;
    }
   printf("\n Number in decimal =%d", num);

}
```

(6) WAP to print the following pattern:

```
5
5 4
5 4 3
5 4 3 2
5 4 3 2 1
```

Ans:-

```
void main()
{
  int i, j
  for(i=5; i>=1; i--)
  {
    for(j=5; j>=i; j--)
    {
      printf("%d", j);
    }
  }
}
```

OK
Sachi