1.Explain about compilaton stages of a c program?write the steps for source code to executable in OS?

Ans:
Compiling a C program is a multi-stage process. At an overview level, the process can be split into four separate stages: Preprocessing, compilation, assembly, and linking. Traditional C compilers orchestrate this process by invoking other programs to handle each stage.
In this post, I'll walk through each of the four stages of compiling the following C program:

```
/*
 * "Hello, World!": A classic.
 */

#include <stdio.h>

int
main(void)
{
        puts("Hello, World!");
        return 0;
}
```

Preprocessing
The first stage of compilation is called preprocessing. In this stage, lines starting with a # character are interpreted by the *preprocessor* as *preprocessor commands*. These commands form a simple macro language with its own syntax and semantics. This language is used to reduce repetition in source code by providing functionality to inline files, define macros and to conditionally omit code.
Before interpreting commands, the preprocessor does some initial processing. This includes joining continued lines (lines ending with a \) and stripping comments.
To print the result of the preprocessing stage, pass the -E option to cc:
cc -E hello_world.c
Given the "Hello, World!" example above, the preprocessor will produce the contents of the stdio.h header file joined with the contents of the hello_world.c file, stripped free from its leading comment:
[lines omitted for brevity]

```
extern int __vsnprintf_chk (char * restrict, size_t,
        int, size_t, const char * restrict, va_list);
# 493 "/usr/include/stdio.h" 2 3 4
# 2 "hello_world.c" 2

int
main(void) {
 puts("Hello, World!");
 return 0;
}
```

Compilation

The second stage of compilation is confusingly enough called compilation. In this stage, the preprocessed code is translated to *assembly instructions* specific to the target processor architecture. These form an intermediate human readable language.
The existence of this step allows for C code to contain inline assembly instructions and for different *assemblers* to be used.
Some compilers also supports the use of an integrated assembler, in which the compilation stage generates *machine code* directly, avoiding the overhead of generating the intermediate assembly instructions and invoking the assembler.
To save the result of the compilation stage, pass the -S option to cc:
cc -S hello_world.c
This will create a file named hello_world.s, containing the generated assembly instructions.

Assembly

During the assembly stage, an assembler is used to translate the assembly instructions to machine code, or *object code*. The output consists of actual instructions to be run by the target processor.
To save the result of the assembly stage, pass the -c option to cc:
cc -c hello_world.c
Running the above command will create a file named hello_world.o, containing the object code of the program. The contents of this file is in a binary format and can be inspected using hexdump or od by running either one of the following commands:
hexdump hello_world.o
od -c hello_world.o

Linking

The object code generated in the assembly stage is composed of machine instructions that the processor understands but some pieces of the program are out of order or missing. To produce an executable program, the existing pieces have to be rearranged and the missing ones filled in. This process is called linking.
The *linker* will arrange the pieces of object code so that functions in some pieces can successfully call functions in other pieces. It will also add pieces containing the instructions for library functions used by the program. In the case of the "Hello, World!" program, the linker will add the object code for the puts function.
The result of this stage is the final executable program. When run without options, cc will name this file a.out. To name the file something else, pass the -o option to cc:
cc -o hello_world hello_world.c
http://www.geeksforgeeks.org/compiling-a-c-program-behind-the-scenes/

2.a)What are the possible errors that compiler may generate?

Ans: Run time errors
compiler errors
Linking errors

b)What are compile time errors and run time errors?

Ans:Compile time errors:

Compile errors are those errors that occur at the time of compilation of the program. C compile errors may be further classified as:

Syntax Errors

When the rules of the c programming language are not followed, the compiler will show syntax errors.

For example, consider the statement,

int a,b:

The above statement will produce syntax error as the statement is terminated with : rather than ;

Semantic Errors

Semantic errors are reported by the compiler when the statements written in the c program are not meaningful to the compiler.

For example, consider the statement,

b+c = a;

In the above statement we are trying to assign value of a in the value obtained by summation of b and c which has no meaning in c. The correct statement will be

a=b+c;

Run time errors:

C runtime errors are those errors that occur during the execution of a c program and generally occur due to some illegal operation performed in the program.

Examples of some illegal operations that may produce runtime errors are:

- Dividing a number by zero
- Trying to open a file which is not created
- Lack of free memory space

It should be noted that occurrence of these errors may stop program execution, thus to encounter this, a program should be written such that it is able to handle such unexpected errors and rather than terminating unexpectedly, it should be able to continue operating. This ability of the program is known as **robustness** and the code used to make a program robust is known as **guard code** as it guards program from terminating abruptly due to occurrence of execution errors.

c)Where you will get the linking error? what is the name of it? how you rectify that?

Ans:

Logical errors are the errors in the output of the program. The presence of logical errors leads to undesired or incorrect output and are caused due to error in the logic applied in the program to produce the desired output.

Also, logical errors could not be detected by the compiler, and thus, programmers has to check the entire coding of a c program line by line.

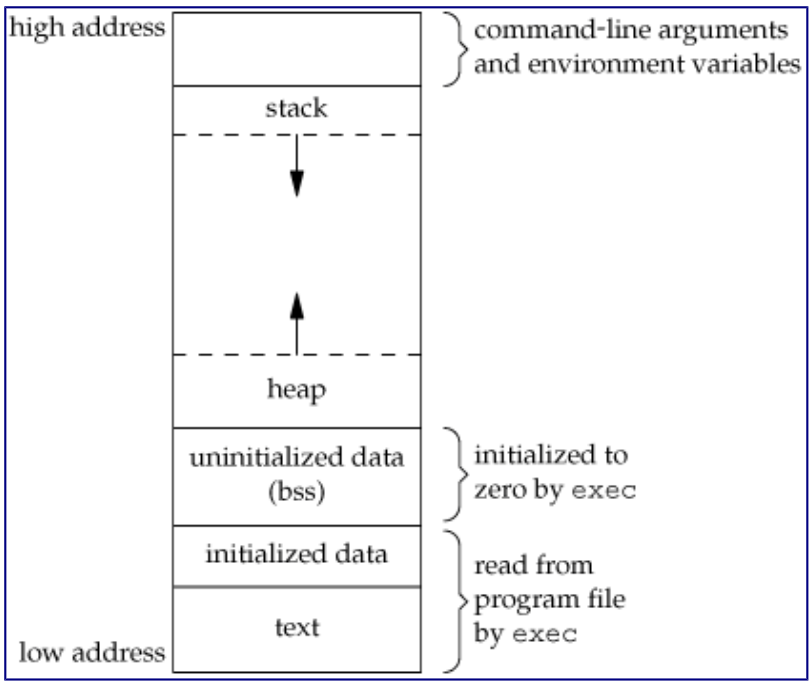
3)Memory layout of a c program?

Ans:

Memory Layout of C Programs

A typical memory representation of C program consists of following sections.

1. Text segment
2. Initialized data segment
3. Uninitialized data segment
4. Stack
5. Heap



A typical memory layout of a running process

1. Text Segment:

A text segment , also known as a code segment or simply as text, is one of the sections of a program in an object file or in memory, which contains executable instructions.

As a memory region, a text segment may be placed below the heap or stack in order to prevent heaps and stack overflows from overwriting it.

Usually, the text segment is sharable so that only a single copy needs to be in memory for frequently executed programs, such as text editors, the C compiler, the shells, and so on. Also, the text segment is often read-only, to prevent a program from accidentally modifying its instructions.

2. Initialized Data Segment:

Initialized data segment, usually called simply the Data Segment. A data segment is a portion of virtual address space of a program, which contains the global variables and static variables that are initialized by the programmer.

Note that, data segment is not read-only, since the values of the variables can be altered at run time.

This segment can be further classified into initialized read-only area and initialized read-write area.

For instance the global string defined by char s[] = "hello world" in C and a C statement like int debug=1 outside the main (i.e. global) would be stored in initialized read-write area. And a global C statement like const char* string = "hello world" makes the string literal "hello world" to be stored in initialized read-only area and the character pointer variable string in initialized read-write area.

Ex: static int i = 10 will be stored in data segment and global int i = 10 will also be stored in data segment

3. Uninitialized Data Segment:

Uninitialized data segment, often called the "bss" segment, named after an ancient assembler operator that stood for "block started by symbol." Data in this segment is initialized by the kernel to arithmetic 0 before the program starts executing

uninitialized data starts at the end of the data segment and contains all global variables and static variables that are initialized to zero or do not have explicit initialization in source code.
For instance a variable declared static int i; would be contained in the BSS segment.
For instance a global variable declared int j; would be contained in the BSS segment.

4. Stack:

The stack area traditionally adjoined the heap area and grew the opposite direction; when the stack pointer met the heap pointer, free memory was exhausted. (With modern large address spaces and virtual memory techniques they may be placed almost anywhere, but they still typically grow opposite directions.)
The stack area contains the program stack, a LIFO structure, typically located in the higher parts of memory. On the standard PC x86 computer architecture it grows toward address zero; on some other architectures it grows the opposite direction. A "stack pointer" register tracks the top of the stack; it is adjusted each time a value is "pushed" onto the stack. The set of values pushed for one function call is termed a "stack frame"; A stack frame consists at minimum of a return address.
Stack, where automatic variables are stored, along with information that is saved each time a function is called. Each time a function is called, the address of where to return to and certain information about the caller's environment, such as some of the machine registers, are saved on the stack. The newly called function then allocates room on the stack for its automatic and temporary variables. This is how recursive functions in C can work. Each time a recursive function calls itself, a new stack frame is used, so one set of variables doesn't interfere with the variables from another instance of the function.

5. Heap:

Heap is the segment where dynamic memory allocation usually takes place.
The heap area begins at the end of the BSS segment and grows to larger addresses from there.The Heap area is managed by malloc, realloc, and free, which may use the brk and sbrk system calls to adjust its size (note that the use of brk/sbrk and a single "heap area" is not required to fulfill the contract of malloc/realloc/free; they may also be implemented using mmap to reserve potentially non-contiguous regions of virtual memory into the process' virtual address space). The Heap area is shared by all shared libraries and dynamically loaded modules in a process.

4)Difference between function pointer and pointer to a function?

Ans:A function pointer is one which returns a pointer.
Ex: int *add(int num1,int num2) {
.
.
}
A pointer to function points to a function.
Ex: int add();
int (*addi)();
addi =&add;
then (*addi)() represents add();
A pointer to a function is a pointer that points to a function. A function pointer is a pointer that either has an indeterminate value, or has a null pointer value, or points to a function.

5)a)Explain about Dynamic Memory Allocation?Differences between malloc() and calloc()functions.

Ans:http://fresh2refresh.com/c-programming/c-dynamic-memory-allocation/

b)How DMA and memcpy are different?

Ans:DMA means direct memory access.In direct memory access without using the cpu we can directly accessing data from the main memory.

Memcpy is just a c library function, which will also copy data from one place to another place by using cpu.

c)You have 5000 line of code and you forgot to free the allocated memory, what will happen.

Ans:Memory Leaks will occur.

6)If have assigned the memory and did not free that and tried to use it again which error will generate.?

Ans:If we are trying to use the same memory in diffrent process ,it will give you run time error,i.e,segmentaution fault.

7)a)What are different sorage data specifiers in c?

Ans:

1.auto
2.register
3.static
4.extern

b)Why we need register storage class?Who will take the decision to store the variable in register?

Ans:

If we declare a variable as a register,then that variable will be stored in CPU registers.

If we want to use a variable frequently,then register storage class will be useful.

--->If we want to use a variable frequently then user should take the decision ,store the variable as a register.

---->If a variable is declared as a register,then compiler should tell that ,strore that variable in CPU register.

c)Is it possible to use a static variable as global variable?

Ans:YES

d)If Yes the same variable we want to use in another source file ?How?

Ans:

The static variable will not be used in two source files,If we want to use one variable in two source files ,then we have to use "extern" storage class.

e)What is extern and when to use it? Explain, differences between static and extern.

Ans:

extern is a keyword,which is used to declare a variable or a function,that can used in many source files.

The **static** storage class is used to declare an identifier that is a local variable either to a function or a file and that exists and retains its value after control passes from where it was declared. This storage class has a duration that is permanent. A variable declared of this class retains its value from one call of the function to the next. The scope is local. A variable is known only by the function it is declared within or if declared globally in a file, it is known or seen only by the functions

within that file. This storage class guarantees that declaration of the variable also initializes the variable to zero or all bits off.
The **extern** storage class is used to declare a global variable that will be known to the functions in a file and capable of being known to all functions in a program. This storage class has a duration that is permanent. Any variable of this class retains its value until changed by another assignment. The scope is global. A variable can be known or seen by all functions within a program.

f)What is static, where the memory is allocated for static variables?
Ans:The **static** storage class is used to declare an identifier that is a local variable either to a function or a file and that exists and retains its value after control passes from where it was declared. This storage class has a duration that is permanent. A variable declared of this class retains its value from one call of the function to the next. The scope is local.
For static variables memory is allocated in data section.
8)What is segment in C? What are memory segments? Explain about BSS Segment? Why we need it?
Ans:
Memory Segments in C:

1. Text Segment
2. Data segment
3. Bss segment
4. Heap
5. Stack

Refer Question3

9)a)Difference between GCC compilation and makefile compilation?
Ans:
Well ... gcc is a compiler, make is a tool to help build programs. The difference is huge. You can never build a program purely using make; it's not a compiler. What make does it introduce a separate file of "rules", that describes how to go from source code to finished program. It then interprets this file, figures out what needs to be compiled, and calls gcc for you. This is very useful for larger projects, with hundreds or thousands of source code files, and to keep track of things like compiler options, include paths, and so on.

https://stackoverflow.com/questions/768373/what-is-the-difference-between-make-and-gcc

b)Explain about makefile? How makefile knows 2 files has modifiled,it has to compile only those 2?
Ans:
Compiling the source code files can be tiring, especially when you have to include several source files and type the compiling command every time you need to compile. Makefiles are the solution to simplify this task.
Makefiles are special format files that help build and manage the projects automatically.

While compiling a file, the make checks its object file and compares the time stamps. If source file has a newer time stamp than the object file, then it generates new object file assuming that the source file has been changed.

10)What is conditional compilation? What are macros and explain their usage?

Ans:

Conditional compilation in c programming language: Conditional compilation as the name implies code is compiled if certain condition(s) hold true. Normally we use if keyword for checking some condition so we have to use something different so that compiler can determine whether to compile the code or not. The different thing is #if.

Conditional compilation is the process of defining compiler directives that cause different parts of the code to be compiled, and others to be ignored. This technique can be used in a cross-platform development scenario to specify parts of the code that are compiled specific to a particular platform.

Macros:

A macro is a name given to a block of C statements as a pre-processor directive. Being a pre-processor, the block of code is communicated to the compiler before entering into the actual coding (main () function). A macro is defined with the preprocessor directive, #define.

The advantage of using macro is the execution speed of the program fragment. When the actual code snippet is to be used, it can be substituted by the name of the macro. The same block of statements, on the other hand, need to be repeatedly hard coded as and when required.

11)Is it Possible to delacre a register variable as a global variable?

Ans: "NO"

12)Write one function pointer,how to call and access a function using a function pointer? Also what to do understand by declaring a function pointers and initialize them?

Ans:

```
#include <stdio.h>
void my_int_func(int x)
{
    printf( "%d\n", x );
}
int main()
{
    void (*foo)(int);
    foo = &my_int_func;

    /* call my_int_func (note that you do not need to write (*foo)(2) ) */
    foo( 2 );
    /* but if you want to, you may */
    (*foo)( 2 );
```

```
    return 0;
}
```
http://www.cprogramming.com/tutorial/function-pointers.html

13)What are call back functions?

Ans:

a **callback** is any executable code that is passed as an argument to other code, which is expected to *call back* (execute) the argument at a given time. This execution may be immediate as in a **synchronous callback**, or it might happen at a later time as in an **asynchronous callback**. In all cases, the intention is to specify a function or subroutine as an entity that is, depending on the language, more or less similar to a variable.

Example:

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

/* The calling function takes a single callback as a parameter. */
void PrintTwoNumbers(int (*numberSource)(void)) {
    int val1= numberSource();
    int val2= numberSource();
    printf("%d and %d\n", val1, val2);
}

/* A possible callback */
int overNineThousand(void) {
    return (rand()%1000) + 9001;
}

/* Another possible callback. */
int meaningOfLife(void) {
    return 42;
}

/* Here we call PrintTwoNumbers() with three different callbacks. */
int main(void) {
    PrintTwoNumbers(&rand);
    PrintTwoNumbers(&overNineThousand);
    PrintTwoNumbers(&meaningOfLife);
    return 0;
}
```

14)How to toggle abit?

Ans:

```
#include<stdio.h>
int main()
{
        int n,bit,t;
```

```
        printf("enter a number,bit:\n");
        scanf("%d %d",&n,&t);
        for(bit =31;bit>=0;bit--)
                printf("%d",(n>>bit)&1);
        printf("\n");
        int res = (n∧(1<<t));
        for(bit =31;bit>=0;bit--)
                printf("%d",(res>>bit)&1);
        printf("\n");
        return 0;
}
```

15)Define structure with functions pointers (file open, file read, file writing) and define a strcture variable and intialize these function pointers with it?

Ans:

16)What is debugging? How to debug an appilication program?

Ans:

Debugging, in computer programming and engineering, is a multistep process that involves identifying a problem, isolating the source of the problem, and then either correcting the problem or determining a way to work around it. The final step of debugging is to test the correction or workaround and make sure it works.

By using gdb or any debugging tool we can debug the program.

17)Eplain about static and global variables? How static and global are different from each other? Draw back of that variables? Can We use static global variable in other file?

Ans:

Static variables have a property of preserving their value even after they are out of their scope! Hence, static variables preserve their previous value in their previous scope and are not initialized again in the new scope.

Syntax:

static data_type var_name = var_value;

Static and global variables are not used in other source files.

If we declare the global variable as extern then it can be used in other source files.

18)What is const variable? Can we changed the variable where we used ? Give example?

Ans:

http://www.geeksforgeeks.org/const-qualifier-in-c/

We can't change the value of a const variable.

Example:

```
#include<stdio.h>
int main()
{
        const int a =10;
        printf("%d\n",a);
        a =20;  //error:assignment of read-only variable 'a'
```

```
        printf("%d\n",a);
}
```

19)Explain AND,OR bitwise operators and CLEAR a bit with programs?

Ans:

& (bitwise AND) Takes two numbers as operand and does AND on every bit of two numbers. The result of AND is 1 only if both bits are 1.

| (bitwise OR) Takes two numbers as operand and does OR on every bit of two numbers. The result of OR is 1 any of the two bits is 1.

TRUTH TABLES:

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

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

CLEAR a Bit:

```
#include<stdio.h>
int main()
{
        int a,bit,i;
        printf("enter a number");
        scanf("%d",&a);
        for(i=31;i>=0;i--)
                printf("%d",(a>>i)&1);
        printf("\n");
        printf("enter bit position:");
        scanf("%d",&bit);
        printf("============clear operation==========\n");
        int res = (a&(~(1<<bit)));
        for(i=31;i>=0;i--)
                printf("%d",(res>>i)&1);

}
```

20)Given const *int a;const *int b =10;Is it given any compiling errors?(Declared in local and global)?

Ans:

Yes It will give a compile time error.

Correct assigning of a variables are like below:

```
#include<stdio.h>
int main()
{
const int* a;
const int* b=(int *) 10;
        printf("%p\n",a);
        printf("%p\n",b);
}
```

21)Difference between pointer to constant and constant pointer?

Ans:

```
int const  *ptr; // ptr is a pointer to constant int
int *const ptr;  // ptr is a constant pointer to int
const int* ptr;
```

declares ptr a pointer to const int type. You can modify ptr itself but the object pointed to by ptr shall not be modified.

```
const int a = 10;
const int* ptr = &a;
*ptr = 5; // wrong
ptr++;    // right
```

While

```
int * const ptr;
```

declares ptr a const pointer to int type. You are not allowed to modify ptr but the object pointed to by ptr.

```
int a = 10;
int *const ptr = &a;
*ptr = 5; // right
ptr++;    // wrong
```

Generally I would prefer the declaration like this which make it easy to read and understand (read from right to left):

22)What are const and volatile keywords? Explain with a code? Differences between const and volatile? Where and why volatile keywords are used? How volatile works?

Ans:

http://www.geeksforgeeks.org/const-qualifier-in-c/

volatile is a qualifier that is applied to a variable when it is declared. It tells the compiler that the value of the variable may change at any time-without any action being taken by the code the compiler finds nearby.

23)How to use a variable in two . c files which was initialIsed in header file?

Ans:

We have to declare the variable as static in header file,and include that header files in two .c files.

24)Can I use one function return type in another function? How?

Ans:

"YES"

```
#include<stdio.h>
int f1(int a)
{
    a++;
    printf("in f1 %d\n",a);
    return a;
}
int f2(int x)
{
    printf("in f2 :%d\n",x);
    return 0;
}
int main()
{
    int num=10;
    f2(f1(num));
    return 0;
}
```

25)How to find the integer is divIsible by 8 or not without using any modulas operator?

Ans:

```
#include<stdio.h>
int main()
{
    int num;
    printf("enter a number:");
    scanf("%d",&num);
    if(((num>>3)<<3)==num)
    {
        printf("divisible\n");
    }
    else
        printf("no");

    return 0;
}
```

26)What is dIfference between definition and declaration give one example?

Ans:

https://stackoverflow.com/questions/1410563/what-is-the-difference-between-a-definition-and-a-declaration

27)How allocation happens for global variables?

Ans:

For global variables memory allocation will happen in data segment i.e,intiailised or bss segment.

If the variable is initialised it will be allocated in intialised data segment,if it is not initialised,them it will be allocated in bss segment there the variable is initialised to zero.

28)What is memory alignment?

Ans:https://www.ibm.com/developerworks/library/pa-dalign/#N1005E

29)If we remove cache memory then can we use volatile or not?

Ans: "Yes",There is no dependency on cache memory for volatile keyword,volatile is used to avoid compiler optimization.

30)Two source files and one header file, header file included in both the files, header file contains int x, will it give compilation or run time?

Ans: "By declaring the variable as static in header it will n't give any compile time error or run time error,if we compiled together also.

31.32bit address 0-4GB i. e 0 to 2 power 32, How many number of processes are running?

Ans: 2 ^ 32-1

32.How to interchange the adjacent bit in bitwise operator?

Ans:

(((num&0xAAAAAAAA)>>1) | ((num&0x55555555)<<1));

33.Write a function to check given number Is divisible by 8 without using mod opertaor?

Ans:

```
if((num&7)==0))
    printf("divisible\n");
else
    printf("no");
```

34.How you will read the register values present in processor?

Ans:

35.What is difference between C and Embedded C?

Ans:http://theembeddedguy.com/2016/06/07/c-vs-embedded-c/

36.Difference between C and Python?

Ans:http://www.rapidprogramming.com/questions-answers/difference-between-c-and-python-c-vs-python-1426

37.Difference between compiler and interpreter. Which is better to use in real time.?

Ans:http://freefeast.info/difference-between/difference-between-compiler-and-interpreter-compiler-vs-interpreter/

38.How you know CPU register is full or not?

Ans:

If we declare a variable as a register,when we r trying to print the address of that register variable it will give error.,because it will store in cpu registers.

If the cpu registers are full ,then it will be stored in main memory.

If it will give the address without any error,then we came to know that cpu register is full.

39.Difference between global and local variable in term of scope, memory?

Ans:

- Global variables can be used anywhere in a computer program. When talking about local variables, it is just a local computer programming or is local to a function.
- The lifetime or scope of a local variable is just within a procedure or a block whereas the scope of a global variable is throughout the program.
- For local variables memory can be allocated in stack,where as for global variables,memory can be allocated in data segment.

40.What is the functionality of linker?

Ans:

In computing, a **linker** or link editor is a computer program that takes one or more object files generated by a compiler and combines them into a single executable file, library file, or another 'object' file.

41.Explain about 'vim' editor. How to replace string in vim editor?

Ans:

https://vim.rtorr.com/

:%s/old/new/g - replace all old with new throughout file

:%s/old/new/gc - replace all old with new throughout file with confirmations

42.Implement left shift operator without using "<<"?

Ans:

```
res =  num *pow(2,bit)
```

43.Open a file for reading/writing and remove duplicate strings.?

Ans:

44.What is memory leak? How you will detect memory leak? What are the precautions for the memory leak?

Ans:

http://www.geeksforgeeks.org/what-is-memory-leak-how-can-we-avoid/

http://www.codeguru.com/cpp/misc/misc/memory/article.php/c3745/Detecting-Memory-Leaks-in-C.htm

Precautions or Rules to avoid memory leak:

1.Always write "free" after malloc.

2.Never, ever,work with the allocated pointer,use a copy.

3.Don't be parsimonious,use more memory.

4.Always carry array length along with you.

5.Be consistent.And save comments.

45.Implement your own strstr().?

Ans:

https://stackoverflow.com/questions/3557178/implementation-of-strstr-function

46. How to create a random numbers in C?

Ans:By using rand and srand functions we can generate random numbers in c.

```
rand()%100;
srand(getpid());
```

47.Explain a structure declaration, structure padding? How we can avoid the padding? (don't use #pragma). What are the differences between structure and union. Give real life example of C Union.

Ans:http://www.geeksforgeeks.org/structures-c/
http://www.geeksforgeeks.org/structure-member-alignment-padding-and-data-packing/
http://www.geeksforgeeks.org/difference-structure-union-c/

48.What is supervisor mode in C.?

Ans:

49.Write a syntax for a function which takes function ptr as a arg, this fptr takes char as arg return type is int function return type is constant pointer

Ans: const * fun(int (*fp)(char));

50.Memory organization diagram in C

Ans:http://www.geeksforgeeks.org/memory-layout-of-c-program/

51.Difference between array and hash?

Ans:

52.What is difference between passing value and reference or call by value and call by reference? How to pass array to function in C? How to pass array using reference.

Ans:
https://stackoverflow.com/questions/13654138/what-exactly-is-the-difference-between-pass-by-reference-in-c-and-in-c

We can't pass entire array to a function as an argument.
We have to pass by reference only.

53.Can we store floating point values in register variable

Ans:https://www.quora.com/Does-the-register-storage-class-hold-float-values-in-C

54.What is the use of break point in debugging?

Ans:https://msdn.microsoft.com/en-us/library/4607yxb0(v=vs.100).aspx

55.What is stack overflow and stack pointer?

Ans:https://stackoverflow.com/questions/26158/how-does-a-stack-overflow-occur-and-how-do-you-prevent-it

A **stack pointer** is a small register that stores the address of the last program request in a **stack**. A **stack** is a specialized buffer which stores data from the top down. As new requests come in, they "push down" the older ones.

56.How to convert floating point to fixed point values. Why it is required.?

Ans:**Fixed Point and need for Fixed Point**

A fixed-point number is a representation of a real number using a certain number of bits of a type for the integer part, and the remaining bits of the type for the fractional part. The number of bits representing each part is fixed (hence the name, fixed-point). An integer type is usually used to store fixed-point values. Fixed-point numbers are usually used in systems which don't have floating point support, or need more speed than floating point can provide. Fixed-point calculations can be

performed using the CPU's integer instructions. A 32-bit fixed-point number would be stored in an 32-bit type such as int. Normally each bit in an (unsigned in this case) integer type would represent an integer value 2∧n as follows:

```
 1    0    1    1    0    0    1    0        = 2∧7 + 2∧5 + 2∧4 + 2∧1 = 178
2∧7  2∧6  2∧5  2∧4  2∧3  2∧2  2∧1  2∧0
```

But if the type is used to store a fixed-point value, the bits are interpreted slightly differently:

```
 1    0    1    1    0    0    1    0        = 2∧3 + 2∧1 + 2∧0 + 2∧-3 = 11.125
2∧3  2∧2  2∧1  2∧0  2∧-1 2∧-2 2∧-3 2∧-4
```

57.What is Stack corruption?
Ans:https://www.go4expert.com/articles/understanding-stack-corruption-c-t27207/
58.How CPU knows to access only particular member in union?
Ans:Based on the data type of that variable.

59.For the function how will you pass complete array and a variable. Is there any difference?
Ans:For array we have to pass by reference.
For value we have to pass either by value or b reference.

60.Different data types sizes and ranges?
Ans:https://www.tutorialspoint.com/cprogramming/c_data_types.htm

61.Without including header file, can you compile the program?
Ans:https://www.quora.com/In-C-can-we-run-a-program-without-header-files
https://stackoverflow.com/questions/17416719/do-i-need-to-compile-the-header-files-in-a-c-program
b)Why do we need to include header file?
https://gcc.gnu.org/onlinedocs/cpp/Header-Files.html

62.What is the Linux command to print lines from 50 to 70 from a file?
Ans:sed -n '50,70p' <filename.txt>
63.What is Little endian and Big endian?Write a generic C program to find out if the target system is bigendian or little-endian?
Ans:http://www.geeksforgeeks.org/little-and-big-endian-mystery/

```
int num = 1;
if(*(char *)&num == 1)
{
    printf("\nLittle-Endian\n");
}
else
{
    printf("Big-Endian\n");
}
```

64.How will you find out the size of integer without using sizeof() operator?
Ans:

```
#include<stdio.h>
int main()
{
    int var,size;
    size = (&var+1);
    int s= &var;
    int res= size-s;
    printf("%d %d %d\n",size,s,res);
}
```

65.What are the bit fields? What is the use of them? Can a union have a bit field? Why?
Ans:http://www.geeksforgeeks.org/bit-fields-c/
66.Can one element in the structure be static? Why?
Ans:https://stackoverflow.com/questions/225089/why-cant-we-initialize-members-inside-a-structure
https://stackoverflow.com/questions/6013373/usage-of-static-within-a-struct-in-c
67.How will you write threadsafe implementation of memcpy() (standard C library function)?
Ans:http://www.geeksforgeeks.org/write-memcpy/
68.What are the side effects of type casting? What is the solution?
Ans:**Side-effects of Type-casting**

Typecasting are of two types-Implicit and Explicit Typecasting where Explicit Typecasting are done by the programmer and implicit typecasting are done by the compiler itself which results in drastic change in the program as

1. Way the Variable that should be accessed is changed**(Memory Access).**
2. **Value Access** from the variable is changed.

Solution:

By Explicit Typecasting we can avoid the side-effects of Typecasting resulted from the Implicit Typecasting done by the compiler for optimization.

69.What is better – passing a pointer of a structure to the function, or the structure itself? Why?
Ans:Passing Pointer to a structure as an argument to function is better and good in Optimising Memory layout of program i.e.., executable and reduces redefinition of structure variable when loading the executable. By passing a pointer to a structure may result in the modification of structure variable values which can be avoided by using "Const" Access specifier.
70.What is the memory fragmentation? How will it happen? What is the way to eliminate them?
Ans:http://ecomputernotes.com/fundamental/disk-operating-system/what-is-fragmentation-different-types-of-fragmentation

https://stackoverflow.com/questions/60871/how-to-solve-memory-fragmentation
71.What is the memory map of the program which is compiled using C compiler?
Ans:http://www.embeddedc.in/p/automotive-basics-part5.html

b)What are the differences between function and a macro? In what situations macro is better?

Ans:http://www.geeksforgeeks.org/macros-vs-functions/

72.If you right-shift the signed number, what will happen to the sign bit?

Ans:Nothing will effect to the signed bit.

73.What is recursive function? What should be considered while writing the recursive functions? What should be taken care of if you write a recursive function? What are the risks if recursive functions are used?

Ans:http://www.sitesbay.com/cprogramming/c-recursion

https://www.programtopia.net/c-programming/docs/recursion-c-programming#

74.What is a function pointer. When and why are they used?

Ans:http://www.geeksforgeeks.org/function-pointer-in-c/

To Declare Constant Function Pointer

```
typedef void (*FP)(void);
const FP array[3] = {&func1, &func2, &func3};
```

75.What are the stack, heap and the data or global area, and how are they different? Which are initialized and which arent? What is the BSS segment and difference between BSS segment and data segment?

Ans: REFER QUESTION No:3

76..Write a macro which converts big endian to little endian?

Ans:

```
#define MSB2LSBDW( x ) (\
        ( ( x & 0x000000FF ) << 24 ) \
        | ( ( x & 0x0000FF00 ) << 8 ) \
        | ( ( x & 0x00FF0000 ) >> 8 ) \
        | ( ( x & 0xFF000000 ) >> 24 ) \
        )
```

77.Explain the difference between const int * and int const *?

Ans:https://stackoverflow.com/questions/1143262/what-is-the-difference-between-const-int-const-int-const-and-int-const

78.Copy n no of bytes from source memory location to destination memory location without using any C library function. Hint: Source and destination can be overlapping.

Ans:http://www.geeksforgeeks.org/write-memcpy/

b)Differentiate between memcpy, memmove and memset.?

Ans:https://stackoverflow.com/questions/1201319/what-is-the-difference-between-memmove-and-memcpy

https://stackoverflow.com/questions/4415910/memcpy-vs-memmove

79.What is a void pointer? When is it used?

Ans:http://www.geeksforgeeks.org/void-pointer-c/

80.Compare and contrast compilers from interpreters.

Ans:REFER QUESTION NO 37

81.How do you generate random numbers in C?

Ans:By using rand and srand functions we can generate random numbers in c.

```
rand()%100;
srand(getpid());
```

82.What is the difference between text files and binary files?

Ans:http://www.codingeek.com/tutorials/c-programming/text-files-vs-binary-files-in-c-programming-language/

83.Define function which takes argument to pass a function pointer as an argument?

Ans:https://stackoverflow.com/questions/1789807/function-pointer-as-an-argument

84.What is the purpose of the keyword typedef?

Ans:https://www.quora.com/What-exactly-does-typedef-do-in-C

85.What happens when a function is called in? (w.r.t Stack)

Ans:

86.List various sections in program's object file?

Ans:http://www.sco.com/developers/gabi/2003-12-17/ch4.sheader.html

Before linking it is called as sections ,At the time of exection it will be called as segments.

87.What will happen to sign bit and vacated bits if the signed number is right shifted?

Ans:If a negative number is right shifted there is no effect on signed bit,vaccated bits will become 1. For possitive numbers vacated bits will become 0.

88.How negative numbers are stored and how to calculate the 2's compliment?

Ans:Positive integers are generally **stored** as simple binary **numbers** (1 is 1, 10 is 2, 11 is 3 and so on). **Negative** integers are **stored** as the two's complement of their absolute value. The two's complement of a positive **number** is, when using this notation, a **negative number.**

https://softwareengineering.stackexchange.com/questions/239036/how-are-negative-signed-values-stored

89.When should we use pointers in a C program?

Ans:https://www.codingunit.com/c-tutorial-how-to-use-pointers

90.What is Dangling pointer?

Ans:http://www.geeksforgeeks.org/dangling-void-null-wild-pointers/

91.How the static variable retains its value between the function calls? What are static functions? What is their use?

Ans:http://www.geeksforgeeks.org/static-variables-in-c/

http://www.geeksforgeeks.org/what-are-static-functions-in-c/

92.How are the unsigned and signed variables are different? What is the difference between signed and unsigned char (or int)? How negative numbers are stored?

Ans:

93.Write differences between pointer and array in C?

Ans:http://www.geeksforgeeks.org/difference-pointer-array-c/

94.Can you subtract pointers from each other? Why would you?

Ans:Yes.

https://stackoverflow.com/questions/3599645/pointer-addition-vs-subtraction

95.How do you use a pointer to a function and When would you use a pointer to a function?

Ans:http://publications.gbdirect.co.uk/c_book/chapter5/function_pointers.html

96.Explain near, far and huge pointers.?

Ans:http://www.geeksforgeeks.org/what-are-near-far-and-huge-pointers/

b)What is the difference between far and near pointers

Ans:https://stackoverflow.com/questions/1749904/what-is-the-difference-between-far-pointers-and-near-pointers

c)When should a far pointer to be used?
Ans:
97.What happens if you free a pointer twice?
Ans:https://stackoverflow.com/questions/2468853/freeing-memory-twice
98.What is NULL pointer? What is the difference between NULL and NUL?
Ans:http://www.geeksforgeeks.org/few-bytes-on-null-pointer-in-c/
https://www.allinterview.com/showanswers/27930/what-is-the-difference-between-null-nul-keywords-in-c.html
99.How do you determine the direction of stack growth ?
Ans:

```
int* foo(void)
{
    int b,*P=&b;
    printf("Address of local variable in function:%p\n",&b);
    return P;
}
int main()
{
int a;
printf("address of main local variable:%p\n",&a);
int *res = foo();
if(res < &a)
    printf("stack is growing down wards\n");
else
    printf("stack is growing upward\n");
```

100.How will you find loop in a linked list?
Ans:/home/dasinram/My_Practice/Assignments/detect_loop.c
101.Count the number of set bits in an integer?
Ans:

```
    while(data)
    {
        count += data &1;
        data >>= 1;
    }
    printf("number of set bits is %d\n",count);
```

102.How negative numbers are stored and How to calculate the 2's compliment?
Ans:REFER QUESTION NO 88
103.Create a custom malloc and free function using linked lists?
Ans:
104.Write a function that determines if a given variable is a power of 2 or not?
Ans:/home/dasinram/My_Practice/Assignments/pwer2.c or 2power.c
105.What is Reentrant code. What is reentrant function. Give one example?
Ans:http://www.geeksforgeeks.org/reentrant-function/
106.What is State Machine. Write a sample state machine which has three states Idle, Calling and Packet states.?
Ans:

PROGRAMS

1.Write a C program to find the whether the stack is growing UP or DOWN?
Ans:/home/dasinram/My_practice/Assignments/stack_grow.c

2.Find the number of bits in the given unsigned integer number?(asked Optimized code)
Ans:while(data)
```
    {
        count += data &1;
        data >>= 1;
    }
    printf("number of set bits is %d\n",count);
```
3.Write a function that determines if a given variable is a power of 2 or not?
Ans:home/dasinram/My_Practice/Assignments/pwer2.c or 2power.c
4.Write program to check given string Is palindrome or not in c?
Ans:home/dasinram/My_Practice/Assignments/str_pallindrome.c
5.What Happens?(text . h) this header file is included in two files (file1. c, file2. c) ?;text.h, static int x = 10
Ans:Nothing will happen.Both files are compiled.
6.What Happens?(text . h) thIs header file is included in two files (file1. c, file2. c) ;text.h;int x ;?
Ans:You will get a compilation error when we are compiling together.
7.Write your own strrev() and strlen() functions?
Ans:
```
#include <stdio.h>
int string_length(char s[])
{
int i;
for (i = 0; s[i]; i++);
return i;
}
void string_reverse(char s1[])
{
int i, j;
for (i = 0, j = string_length(s1)-1; i<j; i++, j--)
{
int tmp = s1[i];
s1[i] = s1[j];
s1[j] = tmp;
}
}
void main()
{
char s[] = "Hello World";
string_reverse(s);
printf("REVERSE: %s\n", s);
```

```
int res = string_length(s);
printf("LENGTH: %d\n", res);
}
```

8.What is the command to print all the . txt files from root directory. ?

Ans: find / -name "*.txt"

Also what is command to print column of listed file.

Ans:find / -name "*.txt" -exec ls -l {} \;

9.Write a program to merge two sorted array in one new array.?

Ans:

```
#include <stdio.h>
void main()
{

    int arr1[] = {1, 5, 9, 10, 15, 20};
    int arr2[] = {2, 3, 8, 13};
    int arr3[200];
    int s1 = sizeof(arr1)/sizeof(arr1[0]);
    int s2 = sizeof(arr2)/sizeof(arr2[0]);
    int s3 = s1 + s2;
    int i, j, k;

    for(i=0;i<s1; i++)
        {
            arr3[i] = arr1[i];
         }
     for(j=0;j<s2; j++)
        {
            arr3[i] = arr2[j];
            i++;
        }
   for(i=0;i<s3; i++)
        {
           for(k=0;k<s3-1;k++)
             {

                if(arr3[k] >= arr3[k+1])
                 {
                   j=arr3[k+1];
                   arr3[k+1]=arr3[k];
                   arr3[k]=j;
                 }
              }
         }

    printf("\nThe merged sorted array is :\n");
    for(i=0; i<s3; i++)
    {
        printf("%d    ", arr3[i]);
    }
```

```
}
```

10.Write a program to sort the array of 5 elements.

Ans:

```
#include<stdio.h>
int main()
{
int arr[] = {8,2,10,0,5};
int i,j,temp;
for(i=0;i<5;i++)
{
    for(j=i+1;j<5;j++)
    {
        if(arr[i]>arr[j])
        {
            temp = arr[i];
            arr[i]= arr[j];
            arr[j]=temp;
        }
    }
}
for(i=0;i<5;i++)
{
printf(" %d ",arr[i]);
}
printf("\n");
}
```

b)Write a program to reverse the elements in an array?

Ans:

```
void rvereseArray(int arr[], int start, int end)
{
int temp;
    while (start < end)
    {
        temp = arr[start];
        arr[start] = arr[end];
        arr[end] = temp;
        start++;
        end--;
    }
}
```

11.Write a program for find large element in a array?

Ans:

```
for(i=0;i<n;i++)
    {
        if(arr[i]>lar)
            lar=arr[i];
    }
    printf("largets element is %d\n",lar);
```

12.Write a C program to check 5th bit using left shIft or right shIft?
Ans:
((1<<5)&num)>0?printf("bit is 1\n"):printf("bit is 0\n");

13.Write a program to swap the string without temp val.?
Ans:

```
char* swap(char* str)
{
    int end = strlen(str) - 1;
    int start = 0;
    while (start < end)
    {
        str[start] ^= str[end];
        str[end] ^= str[start];
        str[start] ^= str[end];
        ++start;
        --end;
    }
    return str;
```

b)write a program to swap two variables without using 3rd variable?
Ans:

```
a = a^b;
b = a^b;
a = a^b;
```

14.Write a program to reverse a string without using temp variable.
Ans:

```
char* reverse(char* str)
{
    int end = strlen(str) - 1;
    int start = 0;
    while (start < end)
    {
        str[start] ^= str[end];
        str[end] ^= str[start];
        str[start] ^= str[end];
        ++start;
        --end;
    }
    return str;
```

15.Given an array of elements and a variable 'r'. Need to print the two elements (let say x,y) from an array such that x + y = r (with min number of iterations===> bitwise operations)?
Ans:

```
 for(i=0;i<10;i++)
    {
        for(j=i+1;j<10;j++)
        {
```

```
            if(arr[i]+arr[j]==sum)
            {
                printf("the pair is number1:%d index %d ,number2:%d index
%d\n",arr[i],i,arr[j],j);
            }
        }
    }
```

16.Write a optimized code (one line code), function takes a num and find it is even or odd number using bitwise operators?

Ans:
```
int even_odd(int number){ return number&1 ? 1 : 0 ;}
```

17.Write a program to find a string in a file?

Ans:
```
    FILE *fp;
    int line_num = 1;
    int find_result = 0;
    char temp[512];
    if((fp = fopen(fname, "r")) == NULL) {
        return(-1);
    }
    while(fgets(temp, 512, fp) != NULL) {
        if((strstr(temp, str)) != NULL) {
            printf("A match found on line: %d\n", line_num);
            printf("\n%s\n", temp);
            find_result++;
        }
        line_num++;
    }
```

18.Write a program to print the string in ascending order?

Ans:
```
for(i=0;str[i];i++)
    {
        for(j=i+1;str[j];j++)
        {
            if(str[j]>str[i])
            {
                temp=str[j];
                str[j]=str[i];
                str[i]=temp;
            }
        }
    }
```

19.Write a program of calculation using function pointers?

Ans:

20.Find second largest number from array with minimum iteration.?

Ans:

21.Write program to check string Is palindrom or not with minimum iteration?

Ans:
```
int is_palindrome(const char *s)
```

```
{
    size_t l = strlen(s);

    if (l == 0) return 0;
    const char *p = s + l - 1;
    while (*s++ == *p-- && s <= p)
        ;
    return s > p;
}
```

22.Write a program to find a largest length palindrome string from a file.?

Ans:

23.How to count the set bits in a binary file? Explanation only?

Ans:

24.Write a program to find sum of N natural numbers with out using any loop?

Ans:

```
int sum(int n)
{
    if(n == 0)
        return 0;
    else
        return n + sum(n-1);

}
```

25.Write a program to print Fibonacci series.?

Ans:

```
int main()
{
   int n, first = 0, second = 1, next, c;
   printf("Enter the number of terms\n");
   scanf("%d",&n);
   printf("First %d terms of Fibonacci series are :-\n",n);
   for ( c = 0 ; c < n ; c++ )
   {
      if ( c <= 1 )
         next = c;
      else
      {
         next = first + second;
         first = second;
         second = next;
      }
      printf("%d\n",next);
   }
   return 0;
}
```

26.Write a program to find the average of prime and odd numbers between 1 to 100.?

Ans:

27.Write a program to remove duplicates in an array without sorting the array.?

Ans:

```
#include<stdio.h>
void main()
{
   int n, a[10], b[10], count = 0, c, d;
   printf("Enter number of elements in array\n");
   scanf("%d",&n);
   printf("Enter %d integers\n", n);
   for(c=0;c<n;c++)
      scanf("%d",&a[c]);
   for(c=0;c<n;c++)
   {
      for(d=0;d<count;d++)
         {
         if(a[c]==b[d])
            break;
      }
      if(d==count)
      {
         b[count] = a[c];
         count++;
      }
   }
   printf("Array obtained after removing duplicate elements\n");
   for(c=0;c<count;c++)
      printf("%d\n",b[c]);
   return 0;
}
```

28.Write a program for given number is multiple by 9 or not. Without using (*) and while loops.?

Ans:

29.Write a C program for avg of array elements in c ? and write the test cases for this program?

Ans:

```
#include <stdio.h>
int main()
{
   int array[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
   int sum, i;
   float avg;
   sum = avg = 0;
   for(i = 0; i < 10; i++)
   {
      sum = sum + array[i];
```

```
    }
    avg = (float)sum / i;
    printf("Average of array values is %.2f\n", avg);
    return 0;
}
```

30.Write a C program for avg of two dimenstional array for each row ? and write the test cases for this program?

Ans:

31.Write a program for string reverse?

Ans:

```
#include <stdio.h>
#include <string.h>
void string_reverse(char s1[])
{
int i, j;
for (i = 0, j = strlen(s1)-1; i<j; i++, j--)
{
int tmp = s1[i];
s1[i] = s1[j];
s1[j] = tmp;
}
}
void main()
{
char s[] = "Hello World";
string_reverse(s);
printf("REVERSE: %s\n", s);
}
```

32.Write a program to count the number of 1s in a program in an efficient way.?

Ans:

33.Swap alternative bits in an integer. Eg: input : 10101010 (assume for 8 bits), output: 0101010?

Ans:

```
#include <stdio.h>
unsigned int swap(unsigned int x)
{
    unsigned int even = x & 0xAAAAAAAA;
    unsigned int odd = x & 0x55555555;
    even >>= 1;
    odd <<= 1;
    return (even | odd);
}

int main()
{
    unsigned int x;
    scanf("%u", &x);
```

```
    printf("Input number: %u \n", x);
    printf("Result: %u \n", swap(x));
    return 0;
}
```

34.Write a program to count no.of 1pairs no.of zero pairs for a given number?

Ans:

35.Write a C program for integer num 5 converted into binary digits and count how many times 1 in the binary digits number?(eg:1001 here 2 ones are there)?

Ans:

```
#include <stdio.h>
void main()
{
    long num, dec, rem, base = 1, bin = 0, count = 0;
    printf("Enter a decimal integer \n");
    scanf("%ld", &num);
    dec = num;
    while (num > 0)
    {
        rem = num % 2;
        if (rem == 1)
        {
            count++;
        }
        bin = bin + rem * base;
        num = num / 2;
        base = base * 10;
    }
    printf("Input number is = %ld\n", dec);
    printf("Its binary equivalent is = %ld\n", bin);
    printf("No.of 1's in the binary number is = %ld\n", count);
}
```

36.How will you read all 4 bytes in an integer of the size of 4 without using bitwise operators? Assume that size of integer is 4?

Ans:

```
    char *var = (char*)&num;
    for(int i=3;i>=0;i--)
    printf("%d ",*(var+i));
```

37.How will you write the macro to set 3 consecutive bits starting at position P in a number N? #define set_bits(N, P) .....

Ans:

```
        #define set(n,p) n|(7<<p)
```

38.Write a C program to take 2 integers as input, if the bits in a and b are different exchange them else retain them?

Ans:Swap the bits is nothing but swap the numbers.

```
for(bit=31;bit>=0;bit--)
{
```

```
int bit1 = (a>>bit)&1;
int bit2 = (b>>bit)&1;
if(bit1==bit2)
continue;
else
{
a = a∧(1<<bit);
b = b∧(1<<bit);
}
}
```

39.Find out number of 1s present at a given memory location?

Ans:

40.Write function which takes (int num1, int pos1, int pos2, int num2), replace num1 bits from pos1 to pos2 with num2 bits from pos1 to pos2?

Ans:

```
#include<stdio.h>
int main()
{
int num1,num2,pos1,pos2,bit;
printf("pos1 must be greaterthan pos2\n");
printf("enter num1,num2 ,pos1,pos2:");
scanf("%d %d %d %d",&num1,&num2,&pos1,&pos2);
for(bit=pos1;bit>=pos2;bit--)
{
int bit1 = (num1>>bit)&1;
int bit2 = (num2>>bit)&1;
if(bit1==bit2)
continue;
else
{
num1 = num1∧(1<<bit);
num2 = num2∧(1<<bit);
}
}
printf("a=%d b=%d\n",num1,num2);
}
```

41.Write function to check given number Is divIsible by 8 without using mod operator?

Ans:

```
if(((num>>3)<<3)==num)
    {
        printf("divisible\n");
    }
    else
        printf("no");
```

42.Write a program to Swap LSNibble with MSNibble?

Ans:((n&0x0000000F)<<28) | ((n&0xF0000000)>>28) |(n&0x0FFFFFF0)

43.Write a code to find present working system is 32bit or 64bit?

Ans:

```
if(sizeof(int *) == 8)
printf("operating system is 64-bit\n");
else
printf("32-bit\n");
```

44.Write a code to print all 1's in a binary?

Ans:

45.Write a C program to find the value of 9th bit and set 11th bit to zero ?(of a 16 bit register)?

Ans:

```
printf("9th bit is %d,11th bit before:%d\n",(a>>9)&1,(a>>11)&1);
int res = (~(1<<11))&a;
printf("11th bit is %d\n",(res>>11)&1);
```

46.Write a macro to find the biggest among 3 variables?

Ans:`#define big(a,b,c) (a>b?(a>c?a:c):(b>c?b:c))`

47.Given some bits, like 0000 1010 perform left shIft and right shIft operations?

Ans:0000 1010 <<1------->0001 0100

0000 1010 >>1------->0000 0101

48.Write function for setting of Nth position bit of given number? Write function for clearing of Nth position bit of given number?

Ans:

```
printf("============clear operation==========\n");
        int res = (a&(~(1<<bit)));
printf("\n==============set operation=========\n");
        res = (1<<bit)|a;
```

49.Write a program to count the number of set bits in an integer?

Ans:

```
while(data)
    {
        count += data &1;
        data >>= 1;
    }
    printf("number of set bits is %d\n",count);
```

50.A nullterminated ASCII string is given, which contains only characters ‘0’ and ‘1’. Write a program to move all the 0s towards the beginning of the string and all the 1s towards the end of the string using only a single traversal. For example, if the given string is “001011011”, expected output will be “000011111”.

Ans:

```
void divide(char *str)
{
        int j,count=0;

        for(int i=0;str[i];i++)
        {
                if(str[i]=='0')
                {
                        count++;
                }
        }
        for(j=0;j<count;j++)
```

```
        {
                str[j]='0';
        }
        for(j=count;str[j];j++)
        {
                str[j]='1';
        }
}
```

51.Write a function to print all 4 bytes in an unsigned integer of the size of 4 without using bitwise operators.

```
void print_bytes(int num)
{
...
...
}
```

Ans: REFER QUESTION NO 36

52.complete the following macro #define CHANGE_BITS(N, S, C) ..... To set the bit at position S and clear the bit at position C. All the positions are to be counted from least significant bit, which is position 0.

Ans:

53.What is wrong with the following code? Explain why.

```
int fun1(int* p)
{
*p = 10;
}
int fun2(int* p)
{
*p = *p + 2;
}
int main()
{
int x ;
x = fun1(&x) + fun2(&x);
}
```

Ans: No compilation or run time error,But it will give garbage value.

54.Observe the following code:

```
#include <stdio.h>

char * fun(void)
{
  char c[] = "VotaryTech";
  char *p = c;
  c++;     //ERROR
  p++;
  p[5] = 65;
  return c;
}
```

```
int main()
{
 char* t =  fun();
 printf("\n%s\n", t);
}
```

What is wrong in this code? Will it compile successfully? Will this code lead to run-time problem if compiled successfully?

Ans:Array can not be incremented.