C Questions

1.

```
int i=300;
int main ()
{int i=200;
{ int i=100;
printf("%d",i);
}
printf("%d",i);
}
```

2.

```
int main()
{
char str[10] = "hello";
str[5] = 'd';
char *s=”world”;
s[2]='p';
printf("%s",str);
printf(“%s”,s);
}
```

3.

```
int main()
{
int i=2;
char *p=(char *)&i;
if(*p==2)
{
printf("little endian\n");
}
else
printf("big end\n");
}
```

4.

```
int main()
{char *p="abc";
char *q="abc123";
while (*p==*q)
printf ("%c %c", *p, *q);
}
```

5.

```
# define scanf "%s is a string"
int main ()
{printf (scanf, scanf);
}
```

6.

```
void fn (int *a, int *b)
{int *t;
t=a;
a=b;
b=t;
}
int main ()
{int a=2;
int b=3;
fn (&a, &b);
printf ("%d %d", a, b);
}
```

7.

```
int main ()
{
int a [10];
printf ("%d", ((a+9) - (a+1)));
}
```

8.

```
int main ()
{
int y, z;
int x=y=z=10;
int f=x;
float ans=0.0;
f *=x*y;
ans=x/3.0+y/3;
printf ("%d %.2f", f, ans);
}
```

9.

```
static int i=5;
int main ()
{
float  sum=0;
do
{
sum+=(1/i*(1.0));
}while (1<i--);
printf("%f",sum);
}
```

10.

```
int main ()
{
int oldvar=25,newvar=-25;
int swap (int, int);
swap (oldvar, newvar);
printf ("Numbers are %d\t %d", newvar, oldvar);
}
int swap (int oldval, int newval)
```

```
{
int tempval=oldval;
oldval=newval;
newval=tempval;}
```

11.

```
int newval (int);
int main ()
{
int ia []={12,24,45,0};
int i;
int sum=0;
for (i=0;ia [i];i++)
{
    sum+=newval (ia [i]);
}
printf ("Sum= %d", sum);
}
int newval (int x)
{
static int div=1;
return (x/div++);
}
```

12.

```
int main ()
{
int var1, var2, var3, minmax;
var1=6;
var2=5;
var3=4;
minmax=(var1>var2)? (var1>var3)? var1: var3:(var2>var3)? var2: var3;
 printf ("%d\n", minmax);
}
```

13.

```
int main ()
{
void pa (int *a, int n);
int arr [5]={5,4,3,2,1};
pa (arr, 5);
}
void pa (int *a, int n)
{
int i;
for (i=0;i<n;i++)
printf ("%d\n", *(a++)+i);
}
```

14.

```
main() {
unsigned int a = 10;
int b = -19;
puts((a+b)>0? "Positive":"Negative");
}
```

15.

```
int main()
{int *p;
char *q;
printf("%d",p+1-p);
printf("%d",q+1-q);
}
```

16.

```
int main()
{
int a=10;
switch(a)
{
case '1':
printf("ONE\n");
break;
case '2':
printf("TWO\n");
break;
defa1ut:
printf("NONE\n");
}
return 0;
}
```

17.

```
int main()
{
int *p = 1;
p++;
printf("%d", p);
}
```

18.

```
int main()
{
char a[6]="avc";
char *b="aaa";
printf("%d  %d",sizeof(a),sizeof(b));
 printf("%d  %d",strlen(a),strlen(b));
}
```

19.

```
#define clrscr() 100
main()
{
clrscr();
34;
int   d;
printf("%d",scanf("%c",&d));
}
```

20.

```
void fun(int,int);
int main()
{fun(1,3);}
void fun(int i,int n)
{ printf("%d",i);
if(i==n)
return ;
fun(i+1,n);
printf("%d",i);}
```

21.

```
#define print(int) printf(“%d”,int) //Line1
main()
{
auto int i=1;//Line 2
print(i);
{int i=2;
print(i);
{ print(i);
int i=10;
i+=1;//Line3
print(i); }
print(i);}
print(i);}
```

22.

```
struct s1{
char c;
int i;
char c2; };
struct s2{
int i;
char c;
char c2; };
main()
{ cout<<sizeof(s1)<<” “<<sizeof(s2); }
```

23.

```
main() {
unsigned a=0xffff;
a=~a;
printf(“%x”,++a);  }
```

24.

```
main() {
struct s{
char *p;  };
static struct s a[]={“RAVI”};
struct s *pp=a;
printf(“\n%s”,++(pp->p)); }
```

25.

```
main() {
abc(((200,100),(300,400))); }
abc(int i) {
```

```
printf(“%d”,i); }
```

26.

```
int *NEXT(register int i) {
int *ipt;
ipt=&i;
ipt++;
return ipt; }
main() {
int j;
printf(“%d”,(NEXT(j))); }
```

a)if address of j=1234,then ans is 1238 c)Garbage value
b)if address j=1234,then ans is 1235 d)compile time error

27.

```
#define PR(a) printf(“%d\t”,(int)(a));
#define PRINT(a) PR(a);
#define FUDGE(k) k+3.14
main() {
int x=2;
printf(“%d”,x);
PRINT(x*FUDGE(2)); }
```

28.

```
main() {
int **p1,(*p2)[10],*p3[10];
printf(“%d%d%d%d%d
%d”,sizeof(p1),sizeof(*p1),sizeof(p2),sizeof(*p2),sizeof(p3),sizeof(*p3));}
```

29.

```
 #define int char
int main()
{
Int i=65;
printf(“ Size of i is :%d”, sizeof(i) );
return 0;
}
```

Answer: Size of i is: 1
Explanation: since the #define replaces the int by char

30.

```
int main()
{
int i=1, j=2;
switch(i)
{
case 1: printf(“Good”);
break;
case j: printf(“Bad”);
break;
}
}
```

Answer: Compile Error: Constant Expression required in function main
Explanation: The case statement can have only constant expression(i.e. we can not use variables

name directly)

31.

```
main()
{
int *p;
void *v;
*p=10;
v=p;
printf(“%d\n”,*v);// Error
}
```

Answer: Cannot dereference void pointer.

32.

```
main()
{
Int  arr[]={1,2,3,4);
Int *p;
p=arr++;//Error
printf(“%d\n”,*p);
}
```

Answer: Name of array is identifier can not be incremented.
Correct way would be p=arr+1;

33.

```
main()
{
Int(*c)[5];
int *a[5];
printf(“%d    %d\n”,sizeof(c),sizeof(a) );
}
```

Answer: 4 20

Explanation : c is a pointer to an array of five integer and a is identifier of array of 5 pointers. Since each pointer is of size 4, so sizeof returns these values.

34.

```
int main()
{
int i;
printf(“%d  ”,scanf(“%d”,&i)); // Value entered is 100
return 0;
}
```

Answer: 1

Explanation: scanf returns number of items successfully read.

35.

```
#define sqr(x)   x*x
int main()
{
int i;
i=64/sqr(4);
printf(“%d \n”,i);
return 0;
}
```

Answer: 64

Explanation: the macro call sqr(4) will be substituted as 4*4, so the expression becomes 64/4*4.

36.
```
main()
{
main();
}
```
Answer: Stack overflow

Explanation: main function call itself recursively each time pushing itself on stack.
This results in stack overflow.

37.
```
main()
{
static int var=5;
printf("%d",var--);
if(var)
main();
}
```
Answer: 5 4 3 2 1

Explanation: When static storage class is given , it is initialized once. The change in value of static variable is retained even between the function calls. Main is also treated like any ordinary function, which can be called recursively.

Subjective Questions Sample:

1. Find maximum element in an array without using if else.
2. Write 3 methods to swap two integers.
3. Write a method a find whether a number is divisible by 7 or not without using / or % operator.
4. Check whether a number is power of 4 or not. (Hint: Use bitwise operator)
5. Find maximum of two numbers without using if-else. Think more than two methods.