--------------------TEST YOUR "C" Skills ----------------------

CHAPTER 1: DECLARATIONS AND INITIALIZATION

Q1> WHAT IS THE O/P?

```
  main()
  {
   char far *s1,*s2;
   printf("%d%d",sizeof(s1),sizeof(s2));
  }
```

answer>4 2

q2>

```
     o/p?
int x=40;
main()
{
int x=20;
printf("%d",x);
}
```

answer>20

q3>o/p?

```
main()
{
 int x=40;
  {
  int x=20;
  printf("%d",x);
   }
 printf("%d",x);
}
```

answer>20 40

q4>is the following statement declaration or defination

```
  extern int x;
```

answer> declaration

q5>

```
 o/p?
 main()
  {
  extern int i;
  i=20;
  printf("%d",sizeof(i));
```

```
 }
```

answer> error,i undefined
because extern int i is a declaration and not defination

q6>is it true that the global variable have many declarations but only one defination?

answer> yes

q7>is it true that the function may have many decalaratins but only one defination?

answer> yes

q8>in the following program where the variable a is geting defined and where it is declared

```
   main()
  {
   extern int a;  /* declaration*/
     printf("%d",a);
  }

int a=12;   /* defination*/
```

q9>wht will be the o/p of above program/

answer>12

q10>what is the difference between declaration and defination of a variable

answer> declaration:-only gives the type,status and nature of variable without reserving any space for the variable
defination;-actual space is reserverd for the variable and some initial value is given.

q11>if the defination of the external variable occurs in the source file before it's use in a
perticular function then there is no need for an external declaration in the function

answer> true

q12>suppose the program is devided in three source files f1,f2,f3 and the variable is defined in file f1 but used in f2 and f3. In such a casewould we need the external declaration for

for the variable in files f2 and f3?

answer>yes

q13>when we mention the prototype of the function ,we are definig it or declaring it?

answer> declaring it

q14>what is the difference between following declarations

```
extern int fun()
int fun();
```

answer> nothing except that that the firat one gives us hint that function fun is probally in another file.

q15>why does the following programreports the redeclaration error of function display()

```
main()
{
   dispaly();
}

void dispaly()
{
 printf("fggagaetaertrt");
}
```

answer> here the function dispay() is called before it is declared .That is why the complier assumes it to be declared as

```
int display();
```

that accept unspecified no of arguments.i.e. undeclared function assumes to return int

on appering the declaration the fun shows that it returns void hence the error

q16>o/p?

```
main()
{
 extern int fun(float);
 int a;
 a=fun(3.14);
 printf("%d",a);
}

int fun(aa)        /* K & R style of function defenation*/
float aa
{
return((int)aa);
}
```

answer> error
     because we have mixed the ansi prototype with k & r style of function defenation
     If we use an ANSI prototype and pass float to the function then it is promoted to double
   the funcction accepts it in to variable of type float hence the type mismatch occurs
      To remady the situation define the function as
    int fun(float aa)
     {
       .....
     }

q17>point error if any

```
  struct emp
  {
    char name[20];
     int age;
  }

fun(int aa)
 {
  int bb;
  bb=aa*aa;
  return(bb);
 }

main()
{
 int a;
a=fun(20);
printf("%d",a);
}
```

answer> missing semicollon at the end of struct
  due to which the function fun assumed to be returning vsr of type struct emp.
  but it returns an int hence the error

q18> If you are to share the variables or functions across several source files how would you enshore that all definications
   and declarations are consistant?

answer> The best arrangement is to place each defination in a revelent .c file , then put an external declaration in a header
   file (.h file) and use #includeto briang the declaration wherever needed
      The .c file which contains the definations should also include the header file, so that the complier can check that
  the defination matches the declaration.

q19>Correct the error

```
   f(struct emp);
     struct emp
       {
         char name[20];
         int age;
       };
    main()
     {
       struct emp e={"Vivek",21}
        f(e);
     }

   f(struct emp ee)
      {
        printf("\n %s %d",ee.name,ee.age);
      }
```

answer> declare the structure before the prototype of f.

q20> Global variables are available to all functions. Does there exist a mechanism by way of which I can make it available
to some and not to others.

answer>NO.

q21>What do you mean by a translation unit

answer> A trnslation unit is a set of source files as seen by the complier and translated as a unit. Generally one .c file
plus all header files mentioned in the #include directives

q22>What wouldbe the output of the following program

```
  main()
  {
    int a[5]={2,3}
     printf("\n %d %d %d",a[2],a[3],a[4]);
  }
```

answer> 0 0 0
if a automatic array is partially initialised then remaiing elements are initialised by 0

q23>o/p

```
main()
 {
  struct emp
   {
     char name[20];
     int age;
     float sal;
```

```
   };

  struct emp e={"vivek"}
  printf("\n %d %f",e.age, e.sal);
 }
```

answer>0 0.000000
  if an automatic structure is partially initialised then remaining elements are initialised bu 0.

q24>Some books sugget that the fillowing defination should be preceded by the word static. Is it correct?

```
  int a[]={2,3,4,12,32}
  struct emp e={"vinod",23}
```

answer> pre ANSI compilers has such requirment but compliers confirming to ANSI standard does not have such requirment.

q25>point out error

```
 main()
 {
  int(*p)()=fun;
  (*p)();
 }

 fun()
  {
    printf("\n Loud and clear");
  }
```

answer> Here we are initialising function pointer to address of the function fun() but during the time of initialisation the function has not been defined. Hence an error
   To eliminate the error add the prototype of function fun() before the declaration of p, as shown bellow;

```
    extern int fun();       or simply
     int fun();
```

q26> point error if any

```
  main()
  {
    union a
     {
       int i;
       char ch[2];
     };
    union a z=512;
    printf("%d %d",z.ch[0],z.ch[1]);
  }
```

answer> In pre-ANSI complier union vriable can not be initialised . ANSI complier permits initialisation of first member of the union

q27>What do you mean by the scope of the variable? what are the 4 differet types of scopes that a variables can have?

answer> Scope indicates the region over which the variable's declaration has an effect. The four kinds of scopes are: file function,block,prototype.

q28> what are different types of linkages?

answer> There are three different types of linkages : external , internal , and none. External linkage means global, non-static variables and functions, internal linkage means static variables and functions with file scope and no linkage means local variables.