1. For the given structure below, declare the variable type, and print their values as well as addresses;


```c
#include <stdio.h>

void main(){
    int la = 345;
    float fb = 4.5;
    char Chvar = 'Z';

    printf("The value of la is %d\n", la);
    printf("The value of fb is %f\n", fb);
    printf("The value of Chvar is %c\n", Chvar);

    printf("The address of la is %u\n", &la); //& : to locate the memory Address
    printf("The address of fb is %u\n", &fb);
    printf("The address of Chvar is %u\n", &Chvar);
}
```

```c
/*2. Declare two integer variable and assign values to them, and print their addresses.
Additionally, Swap the contents of the variables and print their addresses after swap.
State whether the addresses before and after are equal or not.*/
#include <stdio.h>

void main(){
    int a = 12, b = 13;
    printf("The address of a is %u\n", &a);//6422296
    printf("The address of b is %u\n", &b);//6422292
    int c;
    c = a;
    a = b;
    b = c;
    printf("The address of a is %u\n", &a);//6422296
    printf("The address of b is %u\n", &b);//6422292
}
```

```c
/*3. Write the C statement to declare and initialize the pointer variable,p, for the given
structure and display the values of x, y and z with the help of p.*/
 #include <stdio.h>

void main(){
    float x = 6.7, y = 1.2, z = 2.3;
    float *p;
    p = &x;
    printf("x = %.1f\n", *p); //%.1f:floating-point numbers,limited to 1 decimal place
    p = &y;
    printf("y = %.1f\n", *p);
    p = &z;
    printf("z = %.1f\n", *p);
}
```

```c
/*4.Write the C statement to declare and initialize the pointer variables p1, p2 and p3 for the given structure and display the value of x from p1. Also update the value of x to 100 using pointer p3.*/
#include <stdio.h>

void main(){
    int x = 52;
    int *p1, *p2, *p3;
    p1 = p2 = p3 = &x;
    printf("X = %d\n", *p1);//X = 52
    *p3 = 100;
    printf("X = %d\n", *p1);//X = 100
}
```

```c
/*5. Write the C statement to declare and initialize the pointer variable for the given structure and update the values of a, b and c to be incremented by 10 through the pointer variable.*/
#include <stdio.h>

void main(){
    int a = 12, b = 25, c = 18;
    *(&a) += 10;
    *(&b) += 10;
    *(&c) += 10;
    printf("a = %d\n", a);
    printf("b = %d\n", b);
    printf("c = %d\n", c);
}
```

```c
/* Write the C statement to declare and initialize the pointer variables for the given structure and update the values of a, b and c to be incremented by 10 through their respective pointers*/
 #include<stdio.h>

 void main(){
    int a=12 , b=52 ,c=8;
    int *p1=&a; //p will locate to a's location
    int *p2=&b;
    int *p3=&c;

    *p1+=10;
    *p2+=10;
    *p3+=10;

    printf("%d %d %d ", a,b,c);
 }
```

```c
/* 7. Two pointers are pointing to different variables. Write the C statement to find the greater between a, and b using pointer manipulation.*/
 #include<stdio.h>

 void main(){
    int a=52,b=18;
    int *p1=&a;
    int *p2=&b;
    if(*p1>*p2){
        printf("Pointer 1 is greater.");
    }else{
        printf("Pointer 2 is greater.");
    }
 }
```

```c
/*8. Create a program to display the address and value of each element of the given integer array a. Also perform a close observation on the format of the address and the change of address from index 0 to the last index of the array.*/
#include<stdio.h>

void main(){
    int a[5]={10,20,30,40,50};
    int size=sizeof(a)/sizeof(a[0]);
    for(int i=0;i<size;i++){
        printf("Location: %d %u \n",a[i] , &a[i]);
    }
}
```

```c
/*9. Declare the two arrays to hold the values as shown in the given rectangular boxes. Write the equivalent C statement to print their values and addresses through pointer (Hint: an array name is a pointer to the first element in the array).*/
#include<stdio.h>

void main(){
    int arr1[]={10,13,20,33,44};
    int arr2[]={10.2,13.3,20.0,33.3,45.3,89.9};
    int size1 = sizeof(arr1) / sizeof(arr1[0]);
    int size2 = sizeof(arr2) / sizeof(arr2[0]);

    for(int i=0;i<size1;i++){
         printf("Location %d %u\n",arr1 + i,*(arr1 + i));//prints the address of the i-th element
    }
    for (int i = 0; i < size2; i++){
        printf("%u -> %.2f \n",arr1+i,*(arr1 + i));
    }
 }
```

```c
/*10. Write the C statement to declare and initialize the pointer variables for the given structure and display the array content using pointer.*/
#include<stdio.h>
void main(){
    int arr[]={1,2,3,4,5};
    int *p1=&arr[0],*p2=&arr[1],*p3=&arr[2],*p4=&arr[3],*p5=&arr[4];
    printf("%d %d %d %d %d",*p1,*p2,*p3,*p4,*p5);
 }
```

```c
/*  11. Write the C statement to declare and initialize the pointer variable for the given structure and display the array content using the pointer, ptr..*/
#include<stdio.h>
void main(){
  int arr[5]={1,2,3,4,5};
  int *p=&arr[0];  //always point to the memory location
  int size=sizeof(arr)/sizeof(arr[0]);
  for(int i=0;i<size;i++){
    printf("%d ",*(p+i));
  }
}
```

```c
/*12. As array name is a pointer, so modify the assignment ptr=a rather ptr=&a[0]. Write the C statement to declare and initialize the pointer variable for the given structure and display the array content using pointer, ptr.*/
#include<stdio.h>

void main(){
    int arr[]={1,2,3,4,5};
    int *p=arr;
    int size=sizeof(arr)/sizeof(arr[0]);
    for(int i=0;i<size;i++){
        printf("%d",*(p+i));
    }
 }
```

```c
//Trace the o/t
#include<stdio.h>
void main(){
 int m = 10, n = 5;
 int *mp, *np;
 mp = &m;
 np = &n;
 *mp = *mp + *np;//with change in the mp value , m val will also change
 *np = *mp- *np;//*np = 15 - 5 = 10   m = 15, n = 10
 printf("%d %d\n%d %d\n", m, *mp, n, *np);}//15 15    10 10
```

```c
/*Output?*/
#include<stdio.h>
void main(){
    int m = 25, n = 77;
    char c = '*';
```

```
    int *itemp;
    m = &n; //No pointer so no address stored "no cast"
    itemp = m; //itemp is a pointer
    *itemp = c; //without a cast
    *itemp = &c;
}
```

```
/*Simulate the following structure in C to store 55 in a, 105 in b, 89 in c and 68 in d
using their respective pointers. Additionally, find the maximum among a, b, c and d
through pointer manipulation. Finally Store the maximum to the required variable and
display the maximum.*/
 #include<stdio.h>
 void main(){
    int a, b, c, d;
    int *ptr1=&a, *ptr2=&b, *ptr3=&c, *ptr4=&d;

    *ptr1 = 55;
    *ptr2 = 105;
    *ptr3 = 89;
    *ptr4 = 68;

    int arr[4] = {*ptr1, *ptr2, *ptr3, *ptr4};
    int *arrPtr = arr;
    int max = *arrPtr;

    for (int i=0;i<4;i++) {
        if (*(arrPtr+i)>max)
            max =*(arrPtr + i);
    }
    printf("Max: %d\n", max); }
```

```
/* 16. Simulate the following structure in C to find the element sum of the given arrays
a, b, c and d into sumarray using their respective pointers. The 1-D arrays must be
read/scanned through the pointers.*/

#include <stdio.h>

void main() {
    int a[4], b[4], c[4], d[4], sumarray[4];
    int *arrays[] = {a, b, c, d}; // Array of pointers for a, b, c, d
    int *ps = sumarray;

    // Input
    for (int arr = 0; arr < 4; arr++) {
        printf("Enter values for array %c (4 elements):\n", 'a' + arr);
        for (int i = 0; i < 4; i++) {
            scanf("%d", arrays[arr] + i);
        }
    }
    // Calculate
    for (int i = 0; i < 4; i++) {
        *(ps + i) = 0;
```

```
        for (int arr = 0; arr < 4; arr++) {
            *(ps + i) += *(arrays[arr] + i);
        }
    }
    // Output
    printf("\nElement-wise sum of arrays a, b, c, and d:\n");
    for (int i = 0; i < 4; i++) {
        printf("sumarray[%d]: %d\n", i, *(ps + i));
    }
}
```

```
/*17. An argument array is an array of pointers to strings. Write a C Simulation to
implement the following figure and manipulate the character array to hold all capital case
letters using pointer. Finally display the strings.*/
 #include<stdio.h>
 #include <ctype.h>

 void main(){
    char arr[4][10]={"Shubhi","Sahil","Dheeraj","SOA"};//4: element , 10:length
    char *p;
    for(int i=0;i<4;i++){
        p=arr[i];
        while(*p!='\0'){
            if(islower(*p)){
                *p=toupper(*p);
            }
            p++;
        }
    }
    printf("Strings: ");
    for(int i=0;i<4;i++){
        printf("%s\n",arr[i]);
    }
 }
```

19. Write a prototype for a function **sum_n_avg** that has three type double input parameters and two output parameters. The function computes the sum and the average of its three input arguments and relays its results through two output parameters.

Output▼

20. The following code fragment is from a function preparing to call **sum_n_avg** (see question-19). Complete the function call statement.

```
double one, two, three, sum_of_3, avg_of_3;
printf("Enter three numbers> ");
scanf("%lf%lf%lf", &one, &two, &three);
sum_n_avg(_____);
. . .
```

Define the function **sum_n_avg** whose prototype you wrote in question-19.

```c
#include <stdio.h>

void sumNavg(double a, double b, double c, double *sum, double *avg){
    *sum = a+b+c;
    *avg = *sum/3.0;
    }

void main(){
    double one, two, three, sum1, avg1;
    printf("Enter three numbers: ");
    scanf("%lf %lf %lf", &one, &two, &three);
    sum_n_avg(one, two, three, &sum1, &avg1);

    printf("Sum: %.2f\n", sum1);
    printf("Average: %.2f\n", avg1);
}
```

```c
/*21.Write a program to use the idea of multiple calls to a function with input/output
parameters to
 sort 6 integer numbers in ascending order without using any sorting algorithms.*/

#include <stdio.h>
void arrange(int *a, int *b){ //BubbleSort to arrange the element
    if (*a > *b)    {
        int temp = *a;
        *a = *b;
        *b = temp;
    }
}
void main(){
    int nums[6];
    printf("Enter numbers: ");
    for (int i = 0; i < 6; i++) {
        scanf("%d", &nums[i]);
    }
    for (int i = 0; i < 5; i++){
        for (int j = 0; j < 5 - i; j++) {
            arrange(&nums[j], &nums[j + 1]);//sorting
        }
    }
    printf("Ascending Order:  ");
    for (int i = 0; i < 6; i++){
        printf("%d ", nums[i]);
    }
    printf("\n");
}
```

Q21.

```
#include <stdio.h>
void sum(int a, int b, int *cp);
int main(void){
 int x, y, z;
 x = 7; y = 2;
 printf("x y z\n\n");
 sum(x, y, &z);
 printf("%4d%4d%4d\n", x, y, z);
 sum(y, x, &z);
 printf("%4d%4d%4d\n", x, y, z);
 sum(z, y, &x);
 printf("%4d%4d%4d\n", x, y, z);
 sum(z, z, &x);
 printf("%4d%4d%4d\n", x, y, z);
 sum(y, y, &y);
 printf("%4d%4d%4d\n", x, y, z);
 return (0);
}

void sum(int a, int b, int *cp){
  *cp = a + b;
}
```

```
x y z

  7   2   9
  7   2   9
 11   2   9
 18   2   9
 18   4   9
```

Q22.

```
void double_trouble(int *p, int y);

void trouble(int *x, int *y);
int main(void){
int x, y;
trouble(&x, &y);
printf("x = %d, y = %d\n", x, y);//x=13 , y=7
return (0);
}
void double_trouble(int *p, int y){
int x;
x = 10;
*p = 2 * x- y;
}
void trouble(int *x, int *y){
double_trouble(x, 7);
double_trouble(y, *x);
}
```

```
/*24. Develop a program to reverse a string using pointer manipulation.*/
#include <stdio.h>

void main() {
    char str[100]="Shubhangini";
    printf("Reversed string: ");
    int i;
    for (i = 0; str[i] != '\0'; i++);
    for (int j = i - 1; j >= 0; j--) {
        printf("%c", str[j]);
    }
    printf("\n");}
```

```c
/* 25. Design a program to find the largest value in an 1-D array using pointers.*/
#include<stdio.h>
void main(){
    int arr[5]={2,3,4,5,6};
    int *p=arr;
    int max=*p;
    for(int i=0;i<5;i++){
        if (*(p+i) > max)
            max=*(p+i);
    }
    printf("Max: %d",max);
}
```

```c
/*26. Design a program to transpose of a matrix using pointers*/
#include <stdio.h>

void main() {
    int arr[2][3] = {{1, 2, 3}, {4, 5, 6}};
    int res[3][2];
    int *p = (int *)arr;
    int *q = (int *)res;
    // Transpose
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 3; j++) {
            *(q+j*2+i)=*(p+i*3+j);
        }
    }
    //Print
    printf("Transposed matrix:\n");
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 2; j++) {
            printf("%d ",*(q+i*2+j));
        }
        printf("\n");
    }
}
```