1) What is printed?

```c
#include <stdio.h>
int main()
{
   int var = 789;
   int *ptr2;
   int **ptr1;

   ptr2 = &var;
   ptr1 = &ptr2;
   *ptr2 = 12;

   printf("%d %d %d\n", var, *ptr2, **ptr1);

 return 0;
}
```

2) What is printed?

```c
#include <stdio.h>

void fun(int a) {
   printf("Value of a is %d\n", a);
}
int main() {
   void (*fun_ptr)(int) = &fun;
   (*fun_ptr)(10);

   return 0;
}
```

3) What is printed?

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

void afunction(int**x){
   *x = malloc(2 * sizeof(int));
   **x = 12;
   *(*x + 1) = 13;
}

int main(){
   int a = 10;
   int *v = &a;
   afunction(&v);

   printf("%d %d\n", v[0], v[1]);
   return 1;
}
```

4)   What is printed ?

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

int main()
{
    int r = 3, c = 4, i, j, count;

    int **arr = (int **)malloc(r * sizeof(int *));
    for (i=0; i<r; i++)
        arr[i] = (int *)malloc(c * sizeof(int));

    // Note that arr[i][j] is same as *(*(arr+i)+j)
    count = 0;
    for (i = 0; i <  r; i++)
     for (j = 0; j < c; j++)
        arr[i][j] = ++count;  // OR *(*(arr+i)+j) = ++count

    for (i = 0; i <  r; i++)
     for (j = 0; j < c; j++)
        printf("%d ", arr[i][j]);

    /* Code for further processing and free the
       dynamically allocated memory */

    return 0;
}
```

5)   What is printed?

```c
#include <stdio.h>
void SaveValue(int **xPtr, int **yPtr, int arr[])  {
 int i;
 *xPtr=arr + 0;
 *yPtr=arr + 0;
 for(i = 1; i < 5; i++)  {
  if(arr[i] > **xPtr)
     *xPtr = arr + i;
    else if(arr[i] < **yPtr)
     *yPtr = arr + i;
 }
}
int main()  {
   int arr[5]={4, 5, 7, 2, 6};
   int *ptr1;
   int *ptr2;
   SaveValue(&ptr1, &ptr2, arr);
   printf("%d, %d \n", *ptr1, *ptr2);
}
```