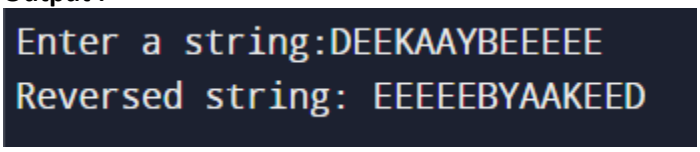
1. Write a program in C to print a string in reverse using a pointer

```
#include <stdio.h>
int main()
{
   int len=0, i=0;
   char str[25], *ptr1, *ptr2;
   printf("Enter a string:");
   gets(str);
   while (str[i] != '\0')
   {
      len = len + 1;
      i++;
   }
   ptr1 = str + len - 1;
   printf("Reversed string: ");
   while (ptr1 >= str)
   {
      printf("%c", *ptr1);
      ptr1--;
   }
   return 0;
}
```

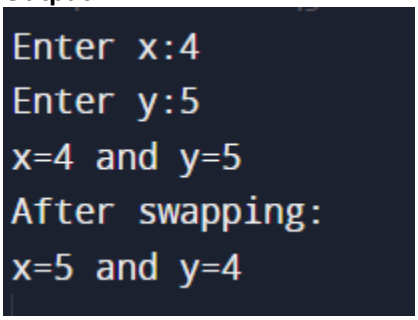
Output :



2. Write a program in C to swap elements using call by reference

```
#include <stdio.h>
void swap(int *x, int *y)
{
   int temp;
   temp = *x;
   *x = *y;
   *y = temp;
}
int main()
{
   int x, y, *ptr1, *ptr2;
   printf("Enter x:");
   scanf("%d", &x);
   printf("Enter y:");
   scanf("%d", &y);
   printf("x=%d and y=%d\n", x, y);
   ptr1 = &x;
   ptr2 = &y;
   printf("After swapping:\n");
   swap(ptr1, ptr2);
   printf("x=%d and y=%d\n", x, y);
   return 0;
}
```

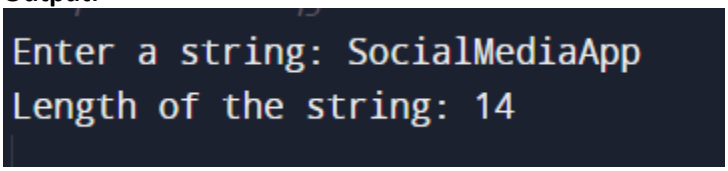
Output:



3. Write a program in C to calculate the length of a string using a pointer.

```
#include <stdio.h>
int len(char *str)
{
   char *ptr = str;
   int length = 0;
   while (*ptr != '\0')
   {
      length++;
      ptr++;
   }
   return length;
}
int main()
{
   char str[25];
   printf("Enter a string: ");
   scanf("%s", str);
   int length = len(str);
   printf("Length of the string: %d\n", length);
   return 0;
}
```

Output:



4. Write a program in C to print all permutations of a given string using pointers.

```
#include<stdio.h>
#include<string.h>
void permu(char *str,int m, int n)
{
   int i,j;
   char tmp;
   for(i = m;i<n-1;++i)
   {
      for(j=i+1;j<n;++j)
      {
         tmp = str[i];
         str[i] = str[j];
         str[j] = tmp;
         permu(str,i+1,n);
         tmp = str[i];
         str[i] = str[j];
         str[j] = tmp;
      }
   }
   printf("%s\n",str);
}
int main()
{
   int i,n;
   char str[25];
   printf("Enter a string:");
   scanf("%s", str);
   while (str[i] != '\0')
   {
      n = n + 1;
      i++;
   }
   permu(str,0,n);
}
```

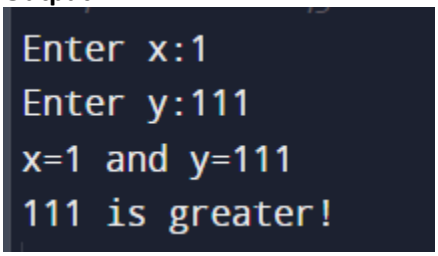
Output:

```
Enter a string:DKB
KBD
KDB
BDK
BKD
DBK
DKB
```

5. Write a program in C to find the maximum number between two numbers using a pointer

```
#include <stdio.h>
void comp(int *x, int *y)
{
   int g;
   if(*x>*y)
      g=*x;
   else
      g=*y;
   printf("%d is greater!\n",g);
}
int main()
{
   int x, y, *ptr1, *ptr2;
   printf("Enter x:");
   scanf("%d", &x);
   printf("Enter y:");
   scanf("%d", &y);
   printf("x=%d and y=%d\n", x, y);
   ptr1 = &x;
   ptr2 = &y;
   comp(ptr1, ptr2);
   return 0;
}
```

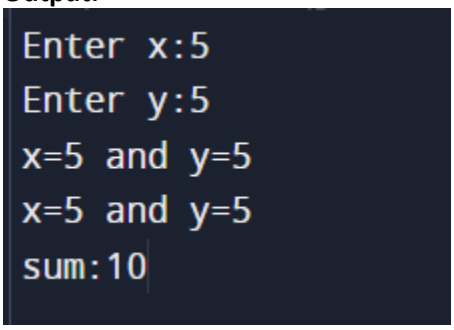
Output:



6. Write a program in C to add numbers using call by reference.

```
#include <stdio.h>
void add(int *x, int *y)
{
   int sum;
   sum= *x+*y;
   printf(“Sum:%d”,sum)
}
int main()
{
   int x, y, *ptr1, *ptr2;
   printf("Enter x:");
   scanf("%d", &x);
   printf("Enter y:");
   scanf("%d", &y);
   printf("x=%d and y=%d\n", x, y);
   ptr1 = &x;
   ptr2 = &y;
   printf("x=%d and y=%d\n", x, y);
   add(ptr1, ptr2);
   return 0;
}
```

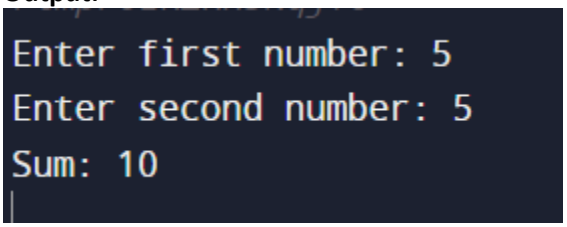
Output:



7. Write a program in C to add two numbers using pointers.

```
#include <stdio.h>
int main()
{
   int x, y, sum;
   int *ptr1, *ptr2;
   printf("Enter first number: ");
   scanf("%d", &num1);
   printf("Enter second number: ");
   scanf("%d", &num2);
   ptr1 = &x;
   ptr2 = &y;
   sum = *ptr1 + *ptr2;
   printf("Sum: %d\n", sum);
   return 0;
}
```

Output:



8. Write a program in C to demonstrate how to handle pointers in a program

```
#include <stdio.h>
int main()
{
   int m;
   int *adr;
   printf("Enter m:");
   scanf("%d", &m);
   printf("Address of m : %p\n", &m);
   printf("Value of m : %d\n", m);
   adr=&m;
   printf("Now adr is assigned with the address of m.\n");
   printf("Address of pointer adr : %p\n", adr);
   printf("Content of pointer adr : %d\n", *adr);
   m = 10;
   printf("The value of m is assigned as 10.\n");
   printf("Address of pointer adr : %p\n", adr);
   printf("Content of pointer adr : %d\n", *adr);
   *adr = 5;
   printf("The pointer variable adr is assigned with the value 5 now.\n");
   printf("Address of m : %p\n", &m);
   printf("Value of m : %d\n", m);
   return 0;
}
```

Output:

```
Enter m:25
Address of m : 0x7ffe82bab524
Value of m : 25
Now adr is assigned with the address of m.
Address of pointer adr : 0x7ffe82bab524
Content of pointer adr : 25
The value of m is assigned as 10.
Address of pointer adr : 0x7ffe82bab524
Content of pointer adr : 10
The pointer variable adr is assigned with the value 5 now.
Address of m : 0x7ffe82bab524
Value of m : 5
```