C POINTER ASSIGNMENT SOLUTION

WWW.learn-in-depth.com

Eng. Keroles Shenouda

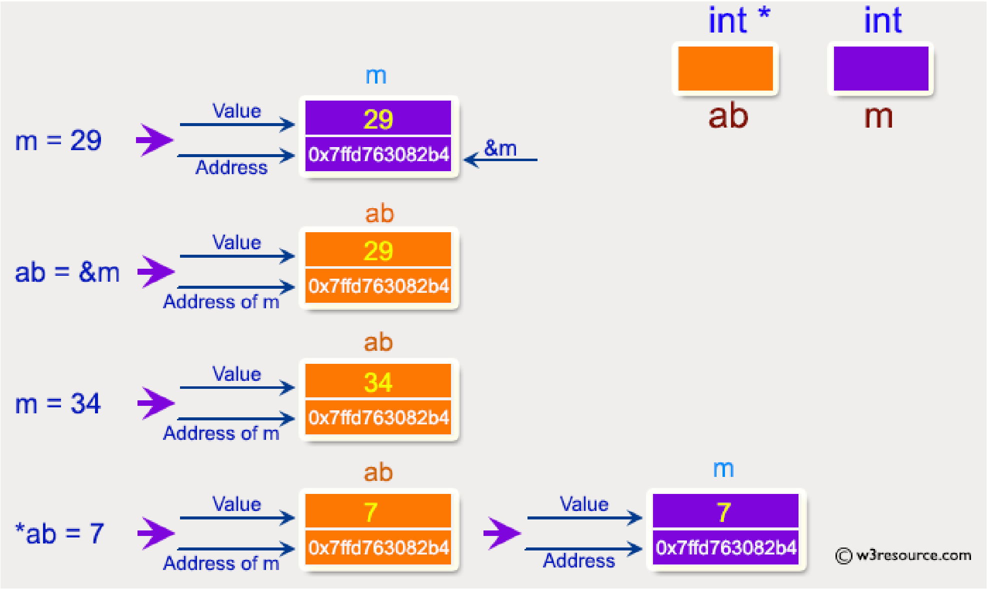
- Q1:

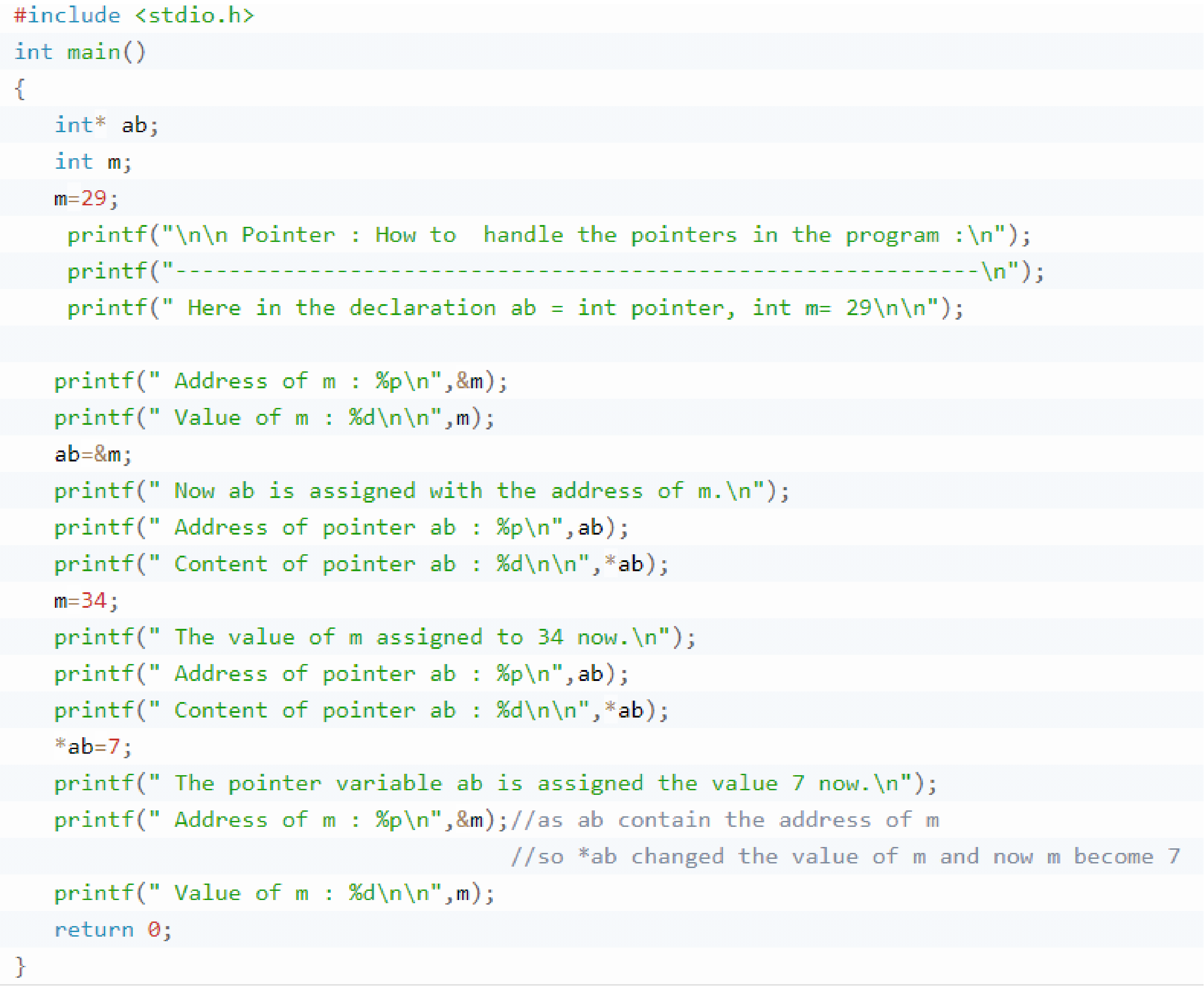
Write a program in C to demonstrate how to handle the pointers in the program.

Expected Output :

```
Address of m : 0x7ffcc3ad291c

 Value of m : 29


 Now ab is assigned with the address of m.

 Address of pointer ab : 0x7ffcc3ad291c

 Content of pointer ab : 29


 The value of m assigned to 34 now.

 Address of pointer ab : 0x7ffcc3ad291c

 Content of pointer ab : 34


 The pointer variable ab is assigned with the value 7 now.

 Address of m : 0x7ffcc3ad291c

 Value of m : 7
```

Solution



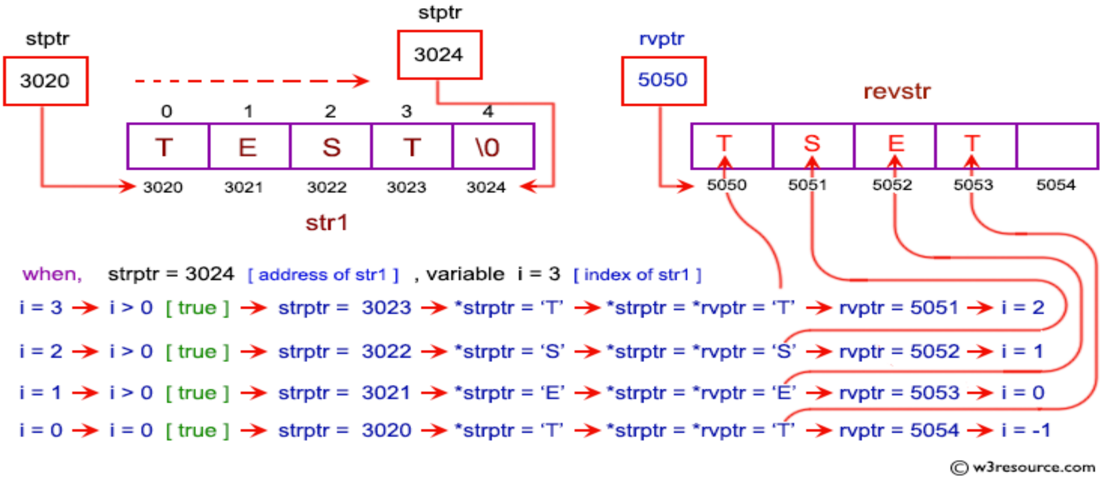


```c
#include <stdio.h>
int main()
{
   int* ab;
   int m;
   m=29;
    printf("\n\n Pointer : How to  handle the pointers in the program :\n");
    printf("------------------------------------------------------------\n");
    printf(" Here in the declaration ab = int pointer, int m= 29\n\n");

   printf(" Address of m : %p\n",&m);
   printf(" Value of m : %d\n\n",m);
   ab=&m;
   printf(" Now ab is assigned with the address of m.\n");
   printf(" Address of pointer ab : %p\n",ab);
   printf(" Content of pointer ab : %d\n\n",*ab);
   m=34;
   printf(" The value of m assigned to 34 now.\n");
   printf(" Address of pointer ab : %p\n",ab);
   printf(" Content of pointer ab : %d\n\n",*ab);
   *ab=7;
   printf(" The pointer variable ab is assigned the value 7 now.\n");
   printf(" Address of m : %p\n",&m);//as ab contain the address of m
                                     //so *ab changed the value of m and now m become 7
   printf(" Value of m : %d\n\n",m);
   return 0;
}
```

■ Q2:

Write a program in C to print all the alphabets using a pointer. Go to the editor

Expected Output :



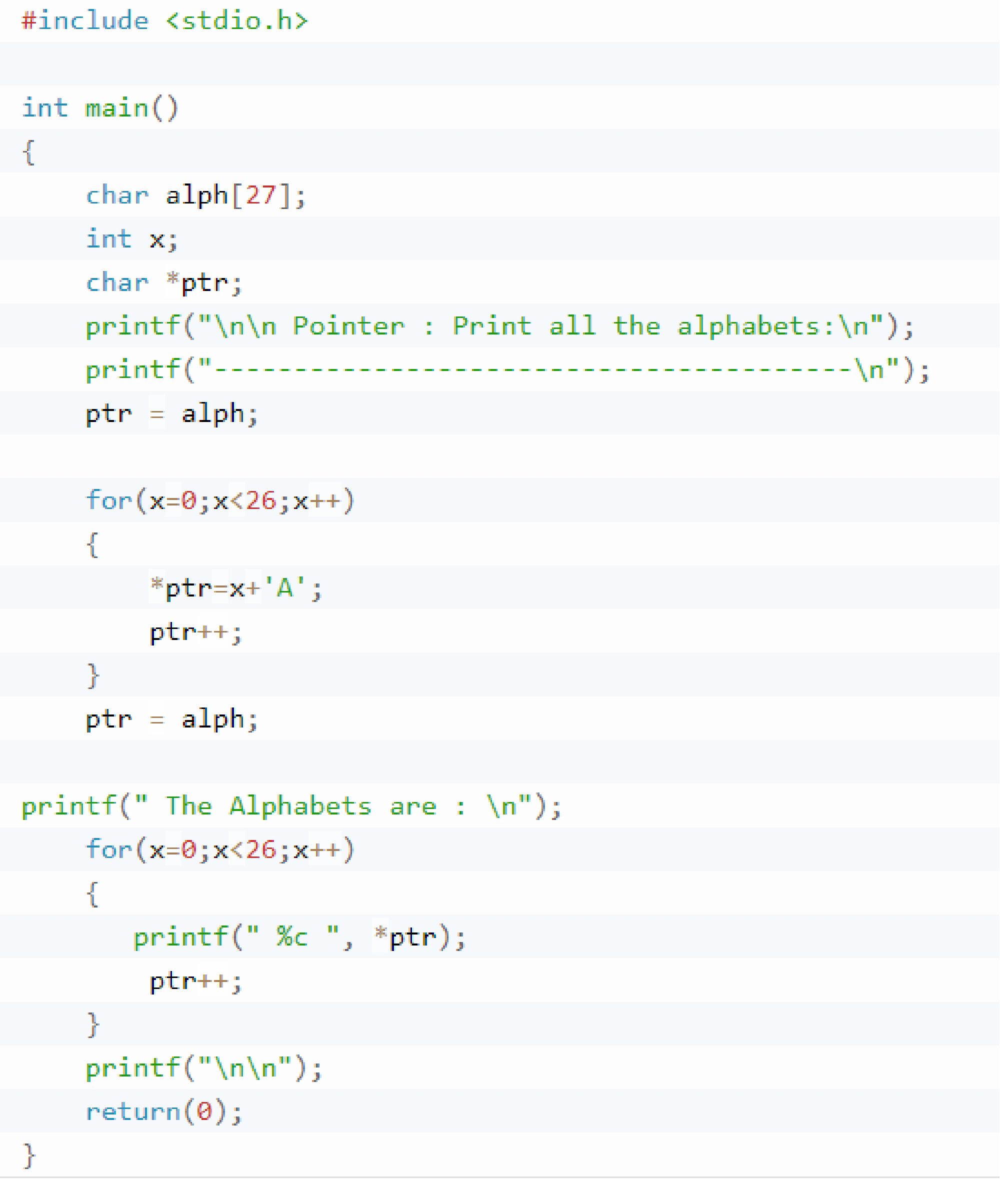
when, strptr = 3024 [address of str1] , variable i = 3 [index of str1]

i = 3 → i > 0 [true] → strptr = 3023 → *strptr = 'T' → *strptr = *rvptr = 'T' → rvptr = 5051 → i = 2

i = 2 → i > 0 [true] → strptr = 3022 → *strptr = 'S' → *strptr = *rvptr = 'S' → rvptr = 5052 → i = 1

i = 1 → i > 0 [true] → strptr = 3021 → *strptr = 'E' → *strptr = *rvptr = 'E' → rvptr = 5053 → i = 0

i = 0 → i = 0 [true] → strptr = 3020 → *strptr = 'T' → *strptr = *rvptr = 'T' → rvptr = 5054 → i = -1



```c
#include <stdio.h>

int main()
{
    char alph[27];
    int x;
    char *ptr;
    printf("\n\n Pointer : Print all the alphabets:\n");
    printf("----------------------------------------\n");
    ptr = alph;

    for(x=0;x<26;x++)
    {
        *ptr=x+'A';
        ptr++;
    }
    ptr = alph;

printf(" The Alphabets are : \n");
    for(x=0;x<26;x++)
    {
       printf(" %c ", *ptr);
        ptr++;
    }
    printf("\n\n");
    return(0);
}
```

- Q2:

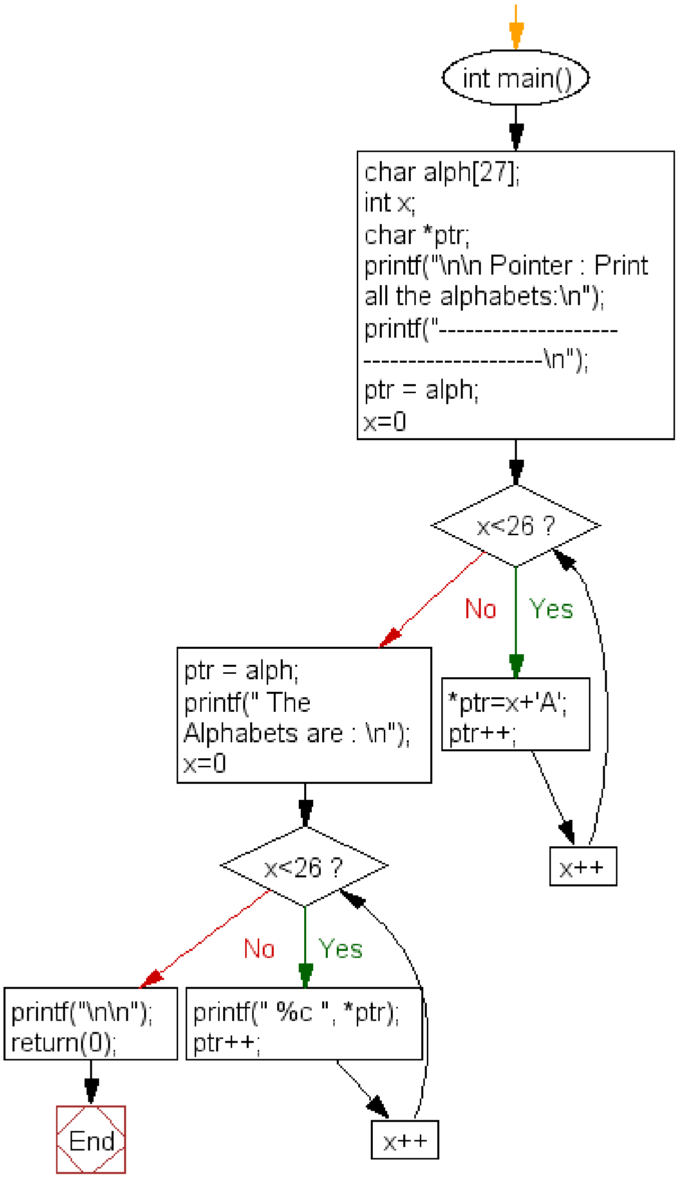
Write a program in C to print all the alphabets using a pointer. Go to the editor

Expected Output :

The Alphabets are :

Expected Output :

```
 A  B  C  D  E  F  G  H  I  J  K  L  M  N  O  P  Q
R  S  T  U  V  W  X  Y  Z
```



```
int main()
```

```
char alph[27];
int x;
char *ptr;
printf("\n\n Pointer : Print all the alphabets:\n");
printf("---------------------------------------\n");
ptr = alph;
x=0
```

```
*ptr=x+'A';
ptr++;
```

```
ptr = alph;
printf(" The Alphabets are : \n");
x=0
```

```
printf(" %c ", *ptr);
ptr++;
```

```
printf("\n\n");
return(0);
```

#LEARN_IN DEPTH
#Be_professional_in_
embedded_system

- Q3:

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

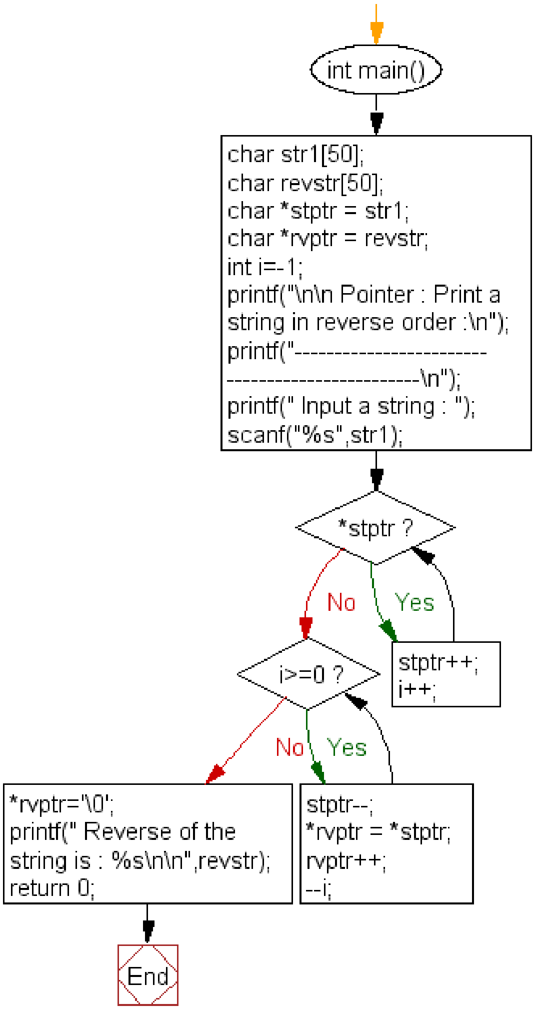
The Alphabets are :

Test Data :

Input a string : w3resource

Expected Output :

```
 Pointer : Print a string in reverse order :
------------------------------------------------
 Input a string : w3resource
 Reverse of the string is : ecruoser3w
```



```
char str1[50];
char revstr[50];
char *stptr = str1;
char *rvptr = revstr;
int i=-1;
printf("\n\n Pointer : Print a
string in reverse order :\n");
printf("------------------------
------------------------\n");
printf(" Input a string : ");
scanf("%s",str1);
```

```
stptr++;
i++;
```

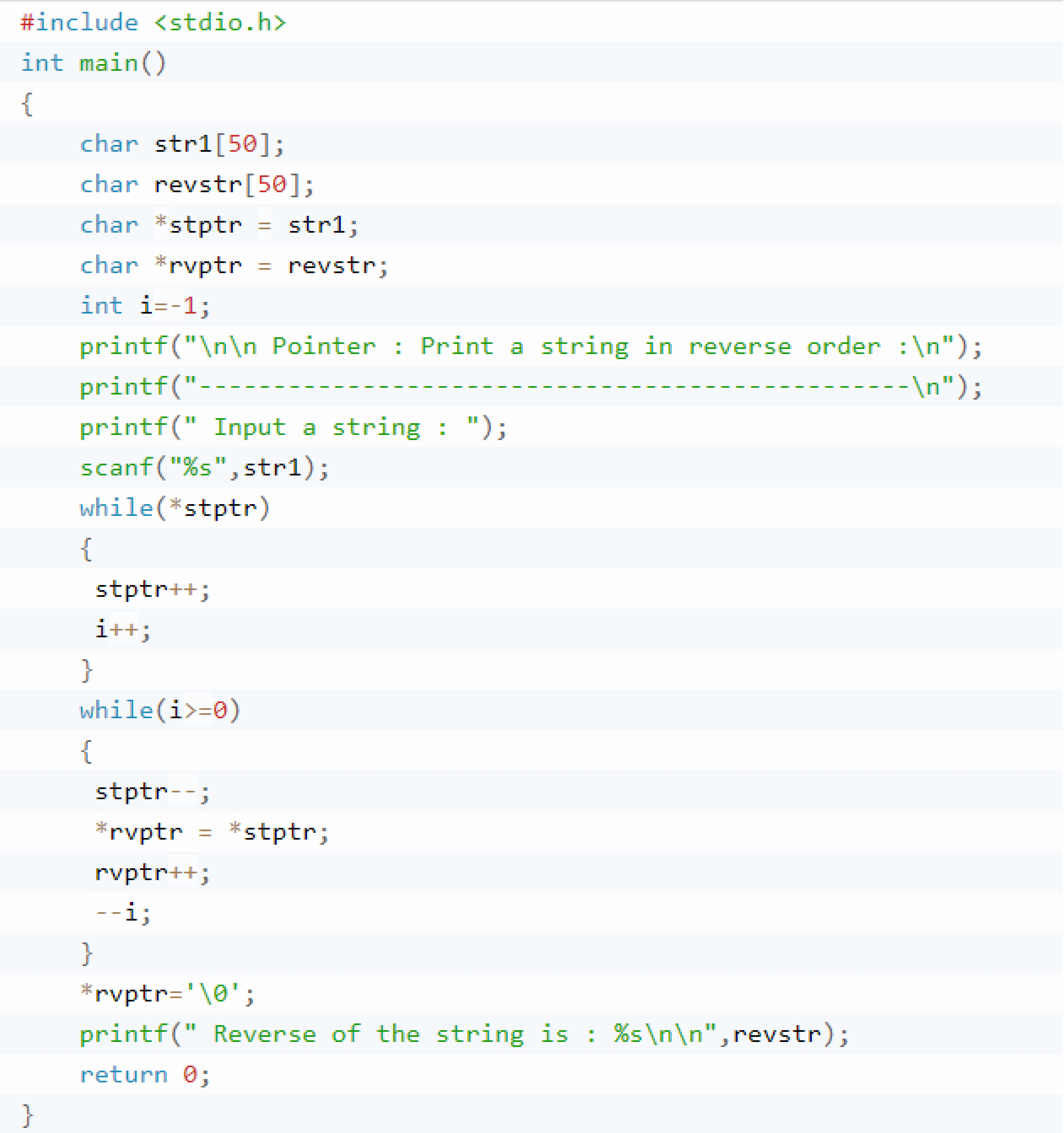
```
stptr--;
*rvptr = *stptr;
rvptr++;
--i;
```

```
*rvptr='\0';
printf(" Reverse of the
string is : %s\n\n",revstr);
return 0;
```

■ Q3:

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

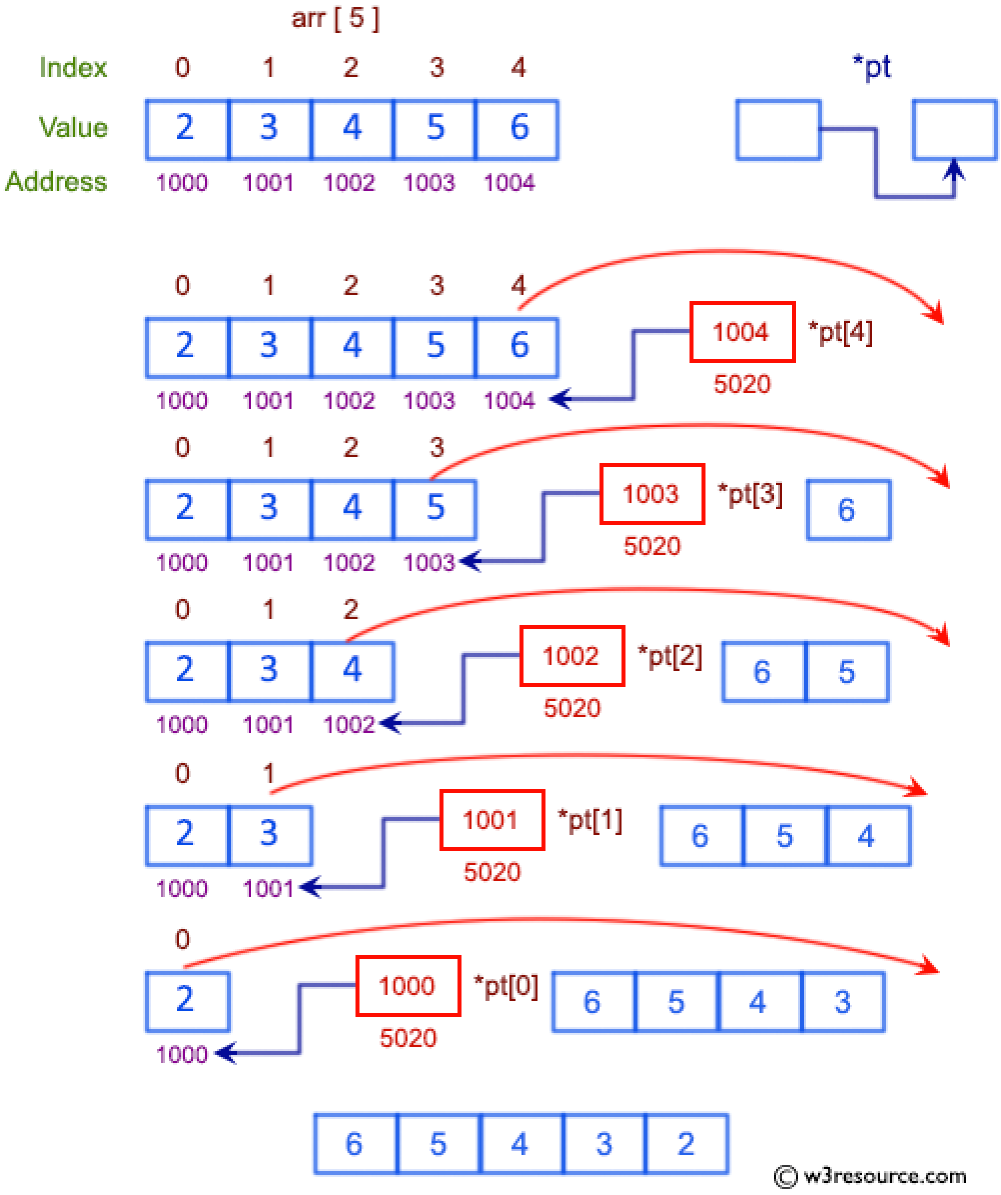
#LEARN_IN DEPTH
#Be_professional_in_
embedded_system



```c
#include <stdio.h>
int main()
{
    char str1[50];
    char revstr[50];
    char *stptr = str1;
    char *rvptr = revstr;
    int i=-1;
    printf("\n\n Pointer : Print a string in reverse order :\n");
    printf("------------------------------------------------\n");
    printf(" Input a string : ");
    scanf("%s",str1);
    while(*stptr)
    {
     stptr++;
     i++;
    }
    while(i>=0)
    {
     stptr--;
     *rvptr = *stptr;
     rvptr++;
     --i;
    }
    *rvptr='\0';
    printf(" Reverse of the string is : %s\n\n",revstr);
    return 0;
}
```

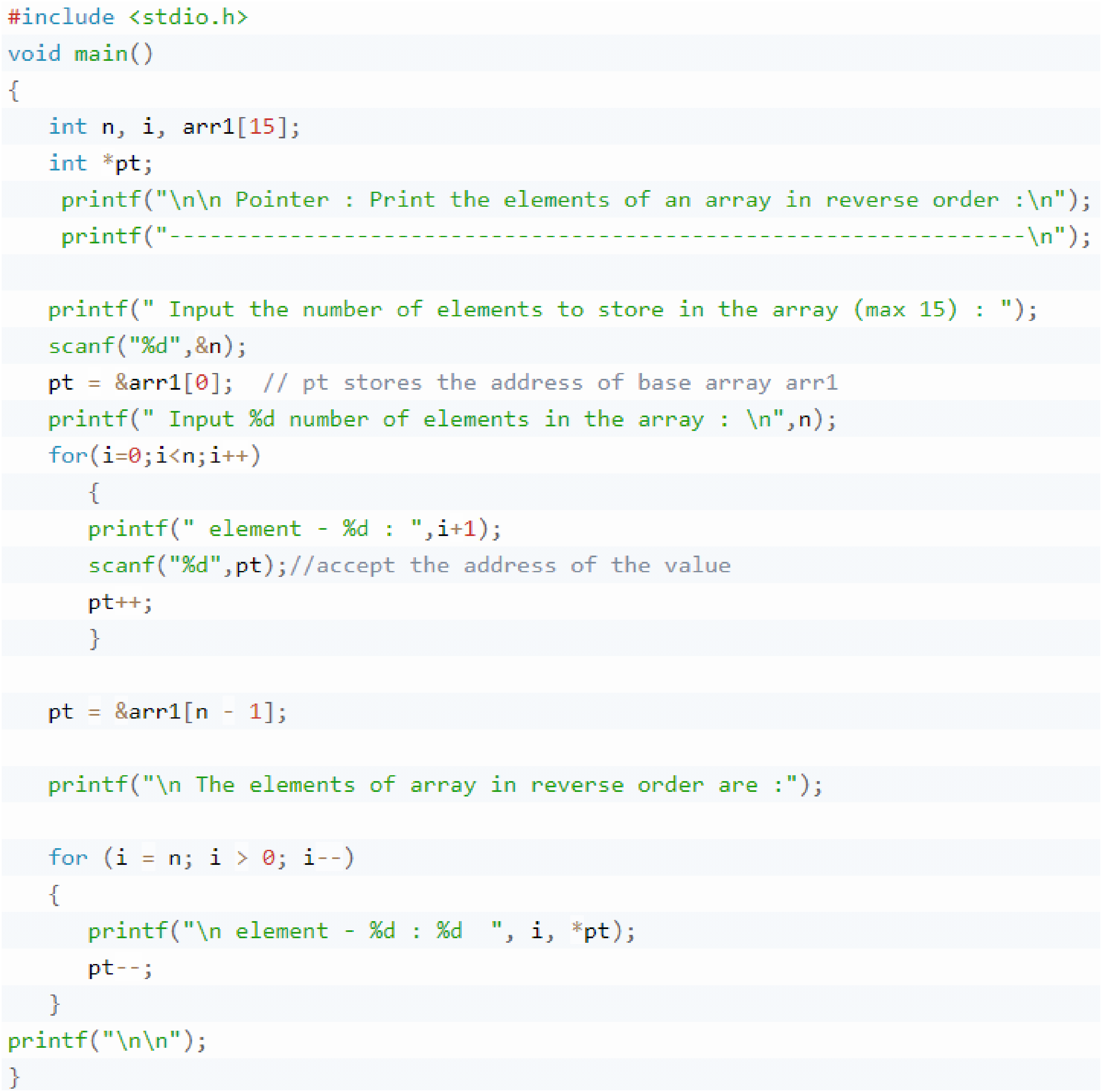
- Q4:

Write a program in C to print the elements of an array in reverse order.



- Q4:

Write a program in C to print the elements of an array in reverse order.



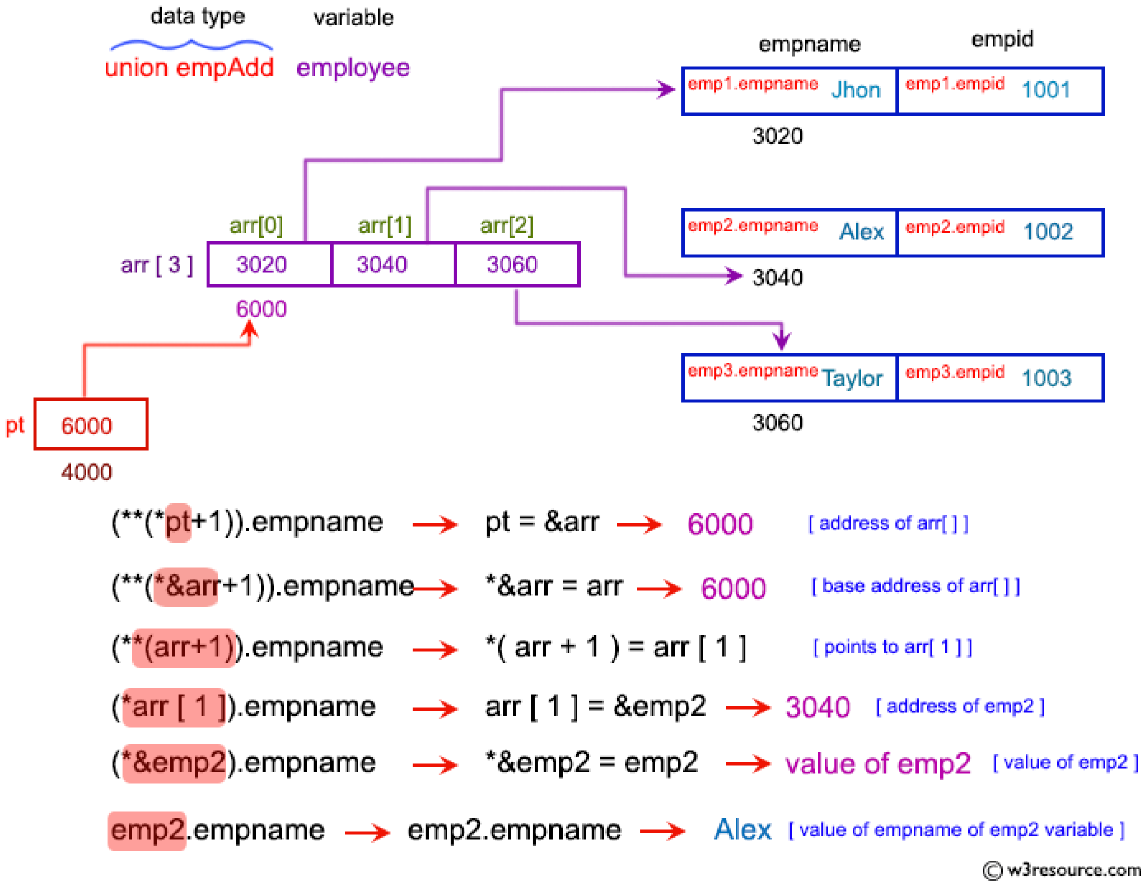
```c
#include <stdio.h>
void main()
{
   int n, i, arr1[15];
   int *pt;
    printf("\n\n Pointer : Print the elements of an array in reverse order :\n");
    printf("----------------------------------------------------------------\n");

   printf(" Input the number of elements to store in the array (max 15) : ");
   scanf("%d",&n);
   pt = &arr1[0];  // pt stores the address of base array arr1
   printf(" Input %d number of elements in the array : \n",n);
   for(i=0;i<n;i++)
      {
      printf(" element - %d : ",i+1);
      scanf("%d",pt);//accept the address of the value
      pt++;
      }

   pt = &arr1[n - 1];

   printf("\n The elements of array in reverse order are :");

   for (i = n; i > 0; i--)
   {
      printf("\n element - %d : %d  ", i, *pt);
      pt--;
   }
printf("\n\n");
}
```

- Q5:

Write a program in C to show a pointer to an array which contents are pointer to structure.

Expected Output :

```
Exmployee Name : Alex
Employee ID :  1002
```


data type
variable
union empAdd
employee
empname
empid
emp1.empname Jhon
emp1.empid 1001
3020
arr[0]
arr[1]
arr[2]
arr [3]
3020
3040
3060
6000
emp2.empname Alex
emp2.empid 1002
3040
emp3.empname Taylor
emp3.empid 1003
3060
pt
6000
4000
(**(*pt+1)).empname → pt = &arr → 6000 [address of arr[ ]]
(**(*&arr+1)).empname → *&arr = arr → 6000 [base address of arr[ ]]
(**(arr+1)).empname → *(arr + 1) = arr [1] [points to arr[1]]
(*arr [1]).empname → arr [1] = &emp2 → 3040 [address of emp2]
(*&emp2).empname → *&emp2 = emp2 → value of emp2 [value of emp2]
emp2.empname → emp2.empname → Alex [value of empname of emp2 variable]
© w3resource.com

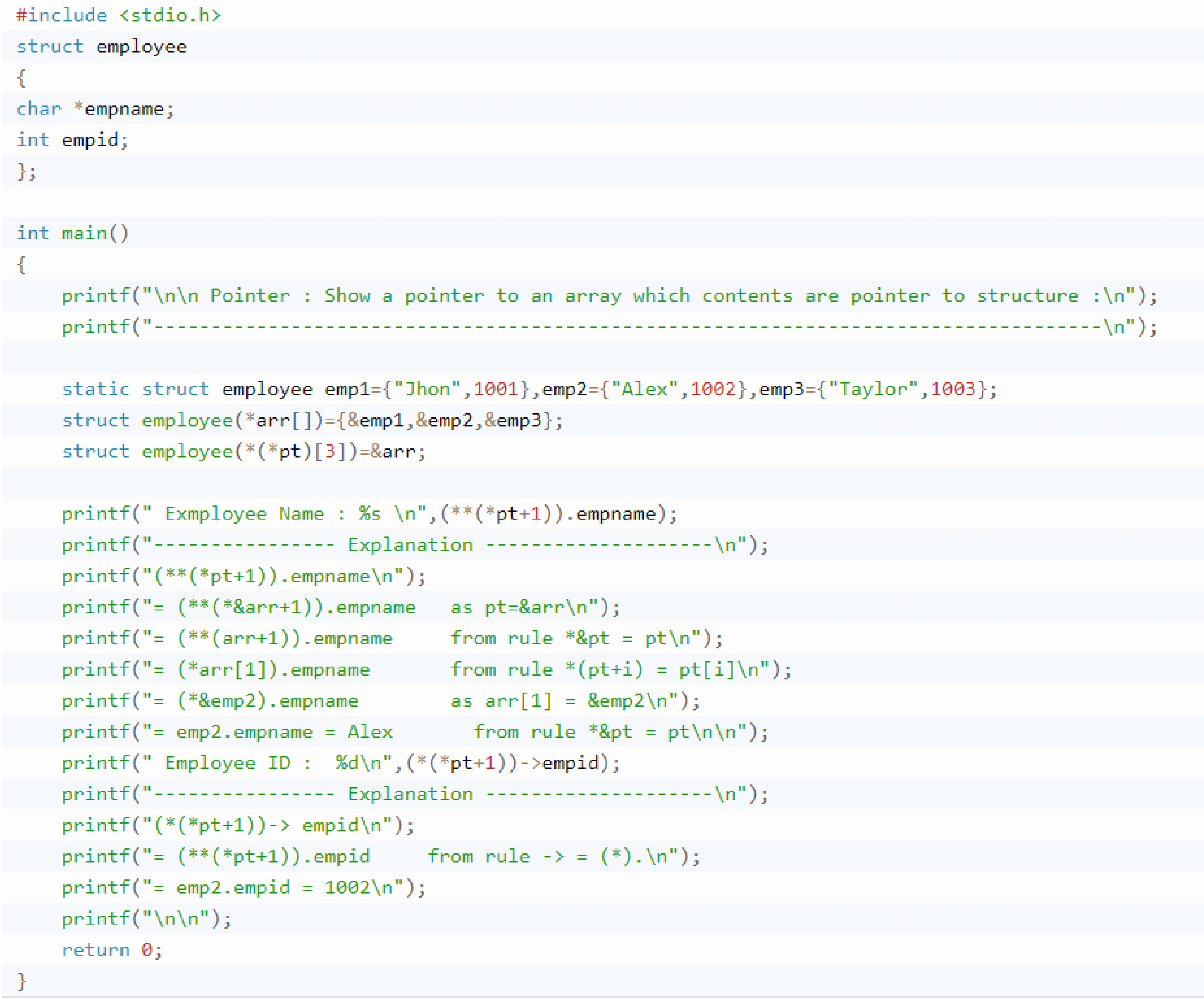
- Q5:

Write a program in C to show a pointer to an array which contents are pointer to structure.

Expected Output :

Exmployee Name : Alex

Employee ID : 1002



```
#include <stdio.h>
struct employee
{
char *empname;
int empid;
};

int main()
{
    printf("\n\n Pointer : Show a pointer to an array which contents are pointer to structure :\n");
    printf("--------------------------------------------------------------------------------------\n");

    static struct employee emp1={"Jhon",1001},emp2={"Alex",1002},emp3={"Taylor",1003};
    struct employee(*arr[])={&emp1,&emp2,&emp3};
    struct employee(*(*pt)[3])=&arr;

    printf(" Exmployee Name : %s \n",(**(*pt+1)).empname);
    printf("---------------- Explanation --------------------\n");
    printf("(**(*pt+1)).empname\n");
    printf("= (**(*&arr+1)).empname   as pt=&arr\n");
    printf("= (**(arr+1)).empname     from rule *&pt = pt\n");
    printf("= (*arr[1]).empname       from rule *(pt+i) = pt[i]\n");
    printf("= (*&emp2).empname        as arr[1] = &emp2\n");
    printf("= emp2.empname = Alex       from rule *&pt = pt\n\n");
    printf(" Employee ID :  %d\n",(*(*pt+1))->empid);
    printf("---------------- Explanation --------------------\n");
    printf("(*(*pt+1))-> empid\n");
    printf("= (**(*pt+1)).empid     from rule -> = (*).\n");
    printf("= emp2.empid = 1002\n");
    printf("\n\n");
    return 0;
}
```