

GDC MEMORIAL COLLEGE

BAHAL (BHIWANI)-127028



DATA STRUCTURE (B.Sc.(cs) 2nd year)

Department of Computer Science

B. Sc. (Computer Science) Semester-IV

List of Programs

Program 1: Calculate and print reverse of an array

Program 2: Find the transpose of a matrix

Program 3: Find the length of given linked list

Program 4: Insert and delete operations over queue

Program 5: Sort n names in the descending order

Program 6: Create encrypted string using the given input string

Program 7: Structure using array

Program 8: Perform push and pop operations over stack in C

Program 9: Sort numeric array using bubble sort

Program 10: Sort n values using quick sort method

/*Program 1: Calculate and print reverse of an array*/

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

```
#include <conio.h >
```

```
void main()
```

```
{
```

```
    int a[10],i,n;
```

```
    clrscr();
```

```
    printf("Enter the size of array\n");
```

```
    scanf("%d",&n);
```

```
    printf("Enter the %d elements\n",n);
```

```
    for(i=0;i < n;i++)
```

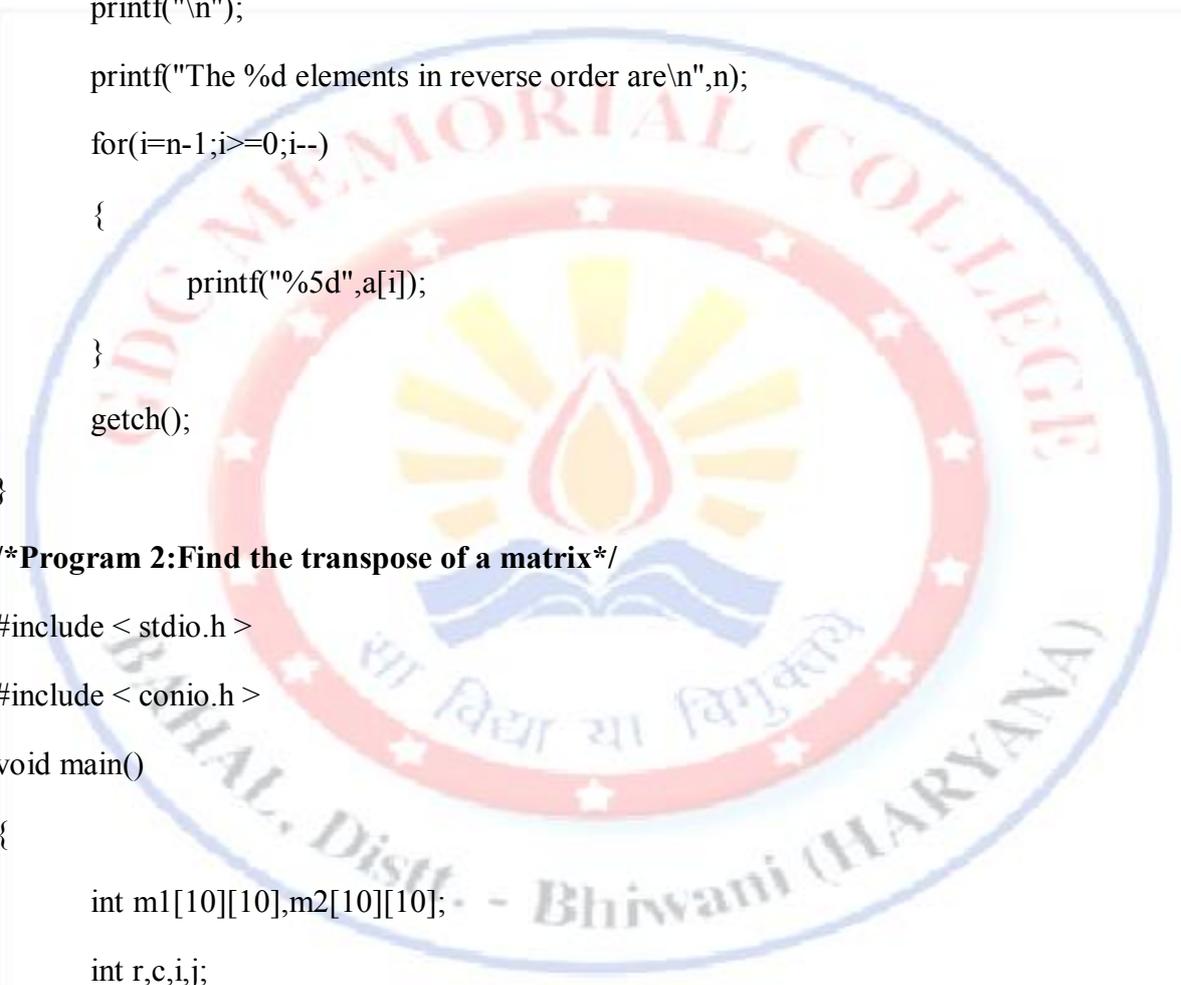
```
    {
```

```
        scanf("%d",&a[i]);
```

```
}
printf("The %d elements are \n",n);
for(i=0;i < n;i++)
{
    printf("%5d",a[i]);
}
printf("\n");
printf("The %d elements in reverse order are\n",n);
for(i=n-1;i>=0;i--)
{
    printf("%5d",a[i]);
}
getch();
}
```

/*Program 2:Find the transpose of a matrix*/

```
#include < stdio.h >
#include < conio.h >
void main()
{
    int m1[10][10],m2[10][10];
    int r,c,i,j;
    clrscr();
    printf("\nEnter r & c : ");
    scanf("%d %d",&r,&c);
    printf("\nEnter first matrix of size %d X %d\n",r,c);
    for(i=0;i< r ; i ++)
```



```

    for(j=0;j < c;j++)
    {
        scanf("%d",&m1[i][j]);
    }
}
for(i=0;i < r ; i++)
{

```

```

    for(j=0;j < c;j++)
    {
        m2[j][i] = m1[i][j];
    }
}
printf("\nthe original matrix is\n");
for(i=0;i < r;i++)
{
    for(j=0;j < c;j++)
    {
        printf("%4d",m1[i][j]);
        printf("\n");
    }
} // for

```

```

printf("\nThe resultane matrix is\n");

```

```

for(i=0;i < c;i++)
{
    for(j=0;j < r;j++)
    {
        printf("%4d",m2[i][j]);
    }
}

```



```
    }
    printf("\n");
} // for
} // main
```

/*Program 3: Find the length of given linked list*/

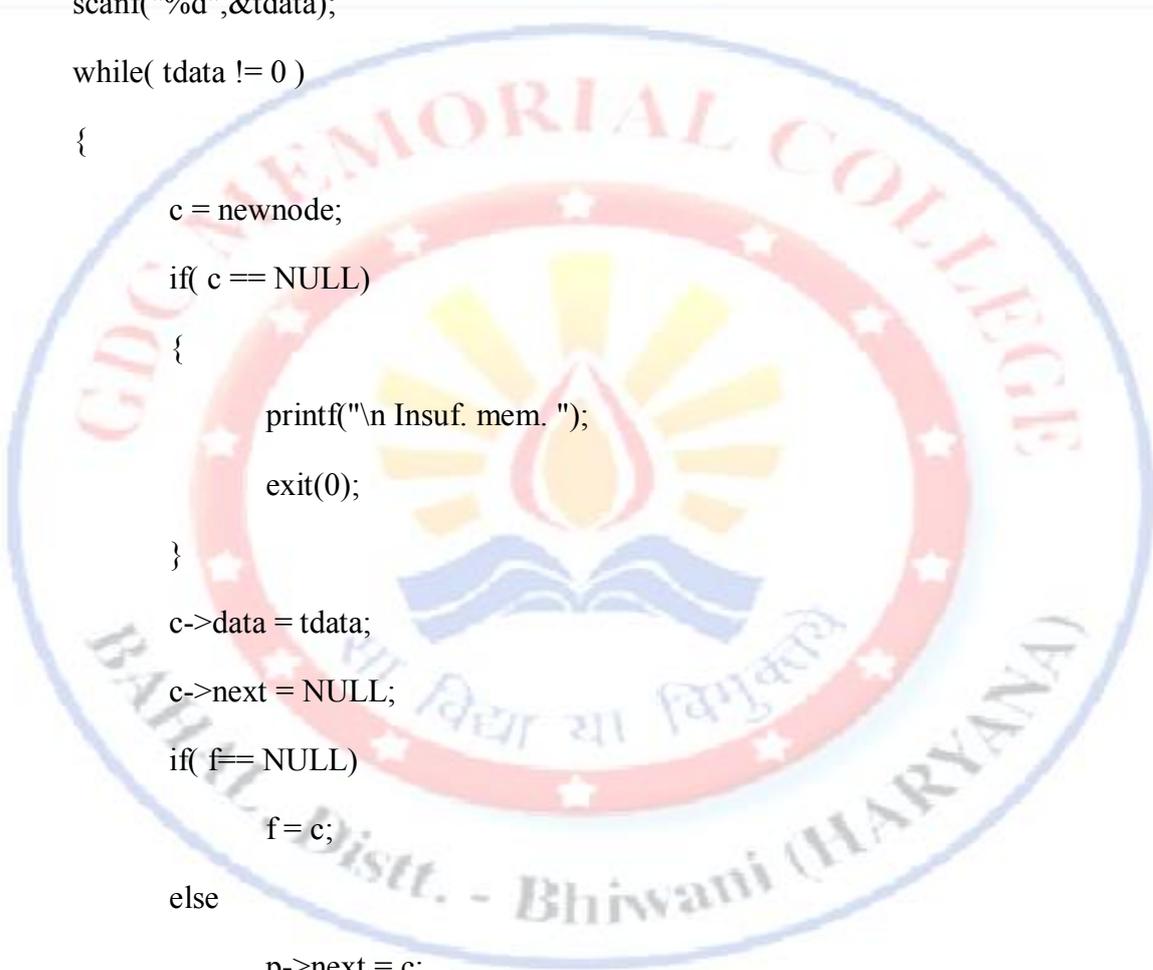
```
#include <stdio.h>
#include <conio.h>
#include <alloc.h>
#define newnode (struct node*) malloc(sizeof(struct node))
struct node
{
    int data;
    struct node *next;
};
struct node *create_list();
void main()
{
    struct node *f;
    int len;
    f = NULL;
    clrscr();

    f = create_list();
    len = find_len(f);
    printf("\n length = %d",len);
} // main
```

```

struct node *create_list()
{
    struct node *f,*c,*p;
    int tdata;
    f = NULL;
    printf("\n Enter data ( use 0 to exit ) : ");
    scanf("%d",&tdata);
    while( tdata != 0 )
    {
        c = newnode;
        if( c == NULL)
        {
            printf("\n Insuf. mem. ");
            exit(0);
        }
        c->data = tdata;
        c->next = NULL;
        if( f== NULL)
            f = c;
        else
            p->next = c;
        p = c;
        printf("\n Enter data ( use 0 to exit ) : ");
        scanf("%d",&tdata);
    } //while
    return(f);
} // create list

```



```

int find_len(struct node *f)
{
    int len=0;
    struct node *t;
    if( f== NULL)
        return(0);
    t = f;
    while( t != NULL )
    {
        len++;
        t = t->next;
    }
    return(len);
}

```

/*Program 4: Insert and delete operations over queue*/

```

#include < stdio.h >
#include < conio.h >
#define max 3
void main()
{
    int queue[max],data;
    int front,rear,reply,option;
    clrscr();
    //... init queue
    front = rear = -1;
    do

```

```
{  
  
    printf("\n 1. insert queue");  
    printf("\n 2. delete queue");  
    printf("\n 3. exit");  
    printf("\nSelect proper option :");  
    scanf("%d",&option);  
    switch(option)  
    {  
        case 1 ://insert  
            printf("\nEnter data : ");  
            scanf("%d",&data);  
            reply = insertq(queue,&rear,&data);  
            if ( reply == - 1 )  
                printf("Queue is full");  
            else  
                printf("Inserted data in queue");  
            break;  
        case 2 : //dele  
            reply = delq(queue,&front,&rear,&data);  
            if ( reply == -1 )  
                printf("Queue is empty ");  
            else  
                printf("Deleted data is : %d", data);  
            break;  
        case 3 : exit(0);  
    } //switch  
} while(1);
```

```

} // main
int insertq ( int queue[max], int *rear , int *data)
{
    if ( *rear == max -1 )
        return(-1);
    else
    {
        *rear = *rear + 1;
        queue[*rear] = *data;
        return(1);
    } // else
} // insert
int delq( int queue[max], int *front, int *rear , int *data)
{
    if ( *front == *rear)
        return(-1);
    else
    {
        (*front)++;
        *data = queue[*front];
        return(1);
    } // else
} // insert

```

/*Program 5: Sort n names in the descending order*/

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

```
#include < conio.h >
```

```
#include < alloc.h >
```

```

void main()
{
    char *list[100],temp[10];
    int n,i,j;
    clrscr();
    printf("How many names?\n");
    scanf("%d",&n);
    printf("Enter %d names \n",n);
    for(i=0;i < n;i++)
    {
        list[i]=(char *)malloc(100);
        fflush();
        gets(list[i]);
    }
    for(i=0;i < n-1;i++)
    {
        for(j=i+1;j < n;j++)
        {
            strcmp(list[i],list[j]);
            {
                strcpy(temp,list[i]);
                strcpy(list[i],list[j]);
                strcpy(list[j],temp);
            }
        }
    }
    printf("The %d names in sorted order are\n",n);

```



```

    for(i=0;i < n;i++)
    {
        puts(list[i]);
    }
} //main

```

/*Program 6: Create encrypted string using the given input string*/

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

```
#include < conio.h >
```

```
void main()
```

```
{
```

```
    char str[100];
```

```
    int diff,new_code,key;
```

```
    int i;
```

```
    clrscr();
```

```
    printf("\nEnter a string \n");
```

```
    gets(str);
```

```
    printf("\nEnter a key : ");
```

```
    scanf("%d",&key);
```

```
    key = key % 27;
```

```
    for(i=0; str[i] != '\0' ; i++)
```

```
    {
```

```
        if ( str[i] >= 'a' && str[i] <= 'z' )
```

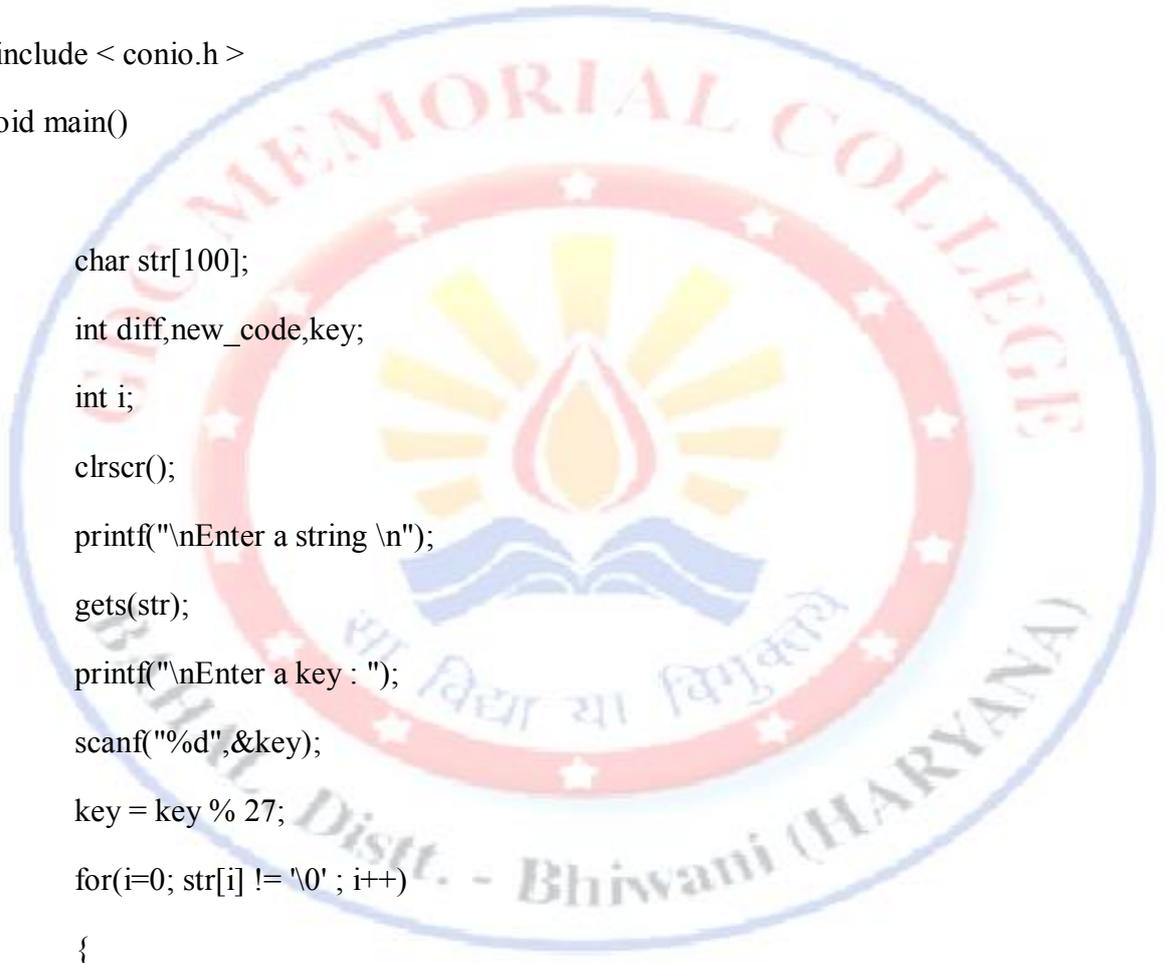
```
        {
```

```
            new_code = str[i] + key;
```

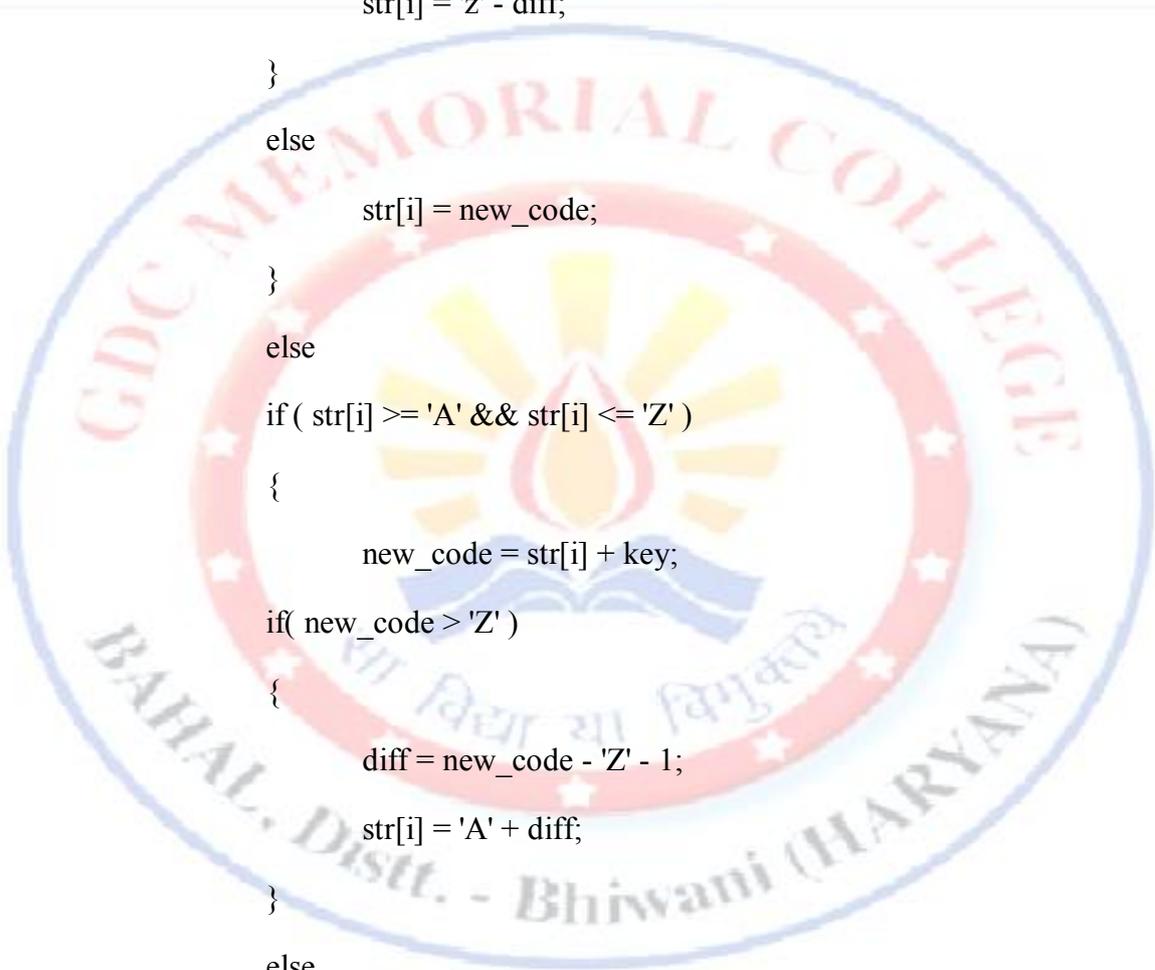
```
            if( new_code > 'z' )
```

```
            {
```

```
                diff = new_code - 'z' - 1;
```



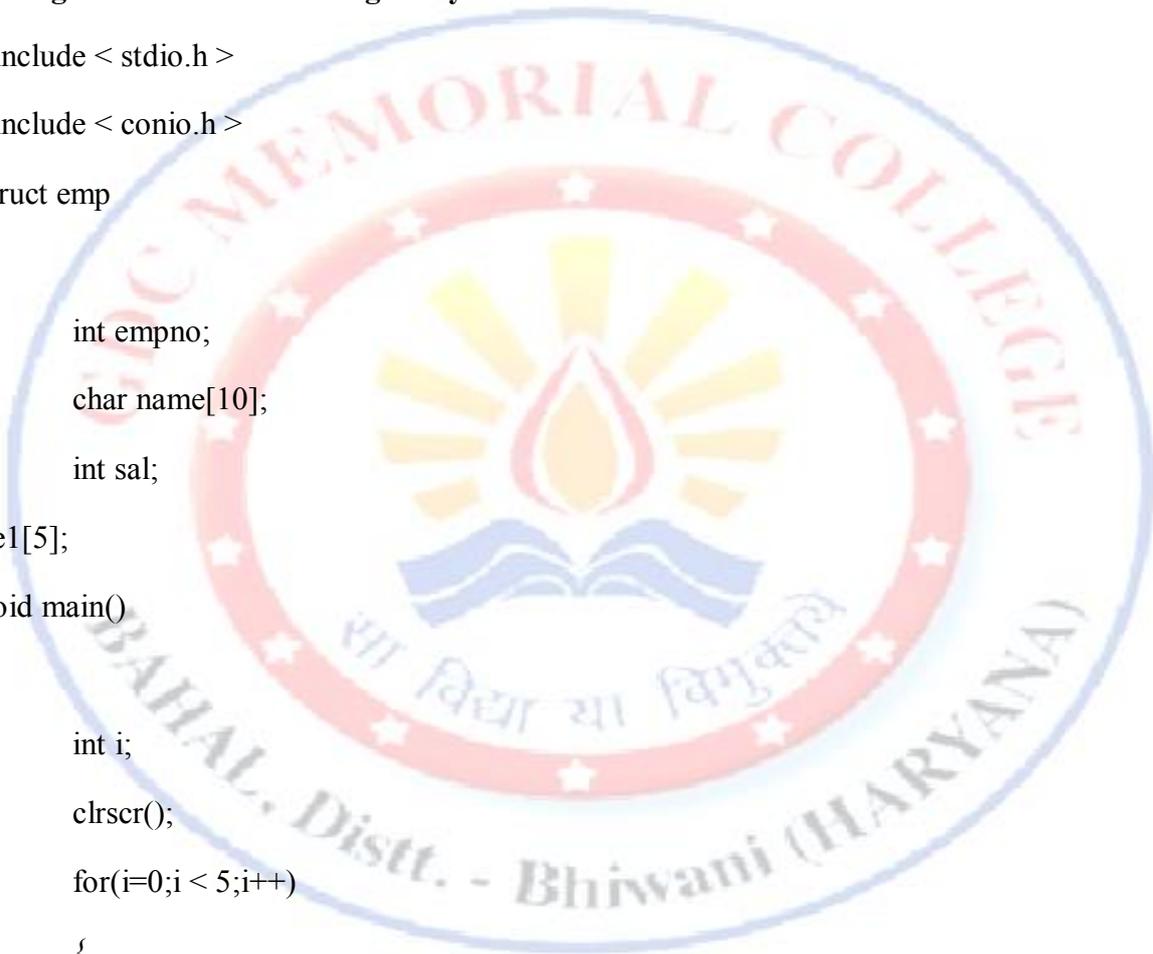
```
        str[i] = 'a' + diff;
    }
    else
    if ( new_code < 'a' )
    {
        diff = 'a' - new_code - 1;
        str[i] = 'z' - diff;
    }
    else
        str[i] = new_code;
    }
    else
    if ( str[i] >= 'A' && str[i] <= 'Z' )
    {
        new_code = str[i] + key;
        if( new_code > 'Z' )
        {
            diff = new_code - 'Z' - 1;
            str[i] = 'A' + diff;
        }
    }
    else
    if ( new_code < 'A' )
    {
        diff = 'A' - new_code - 1;
        str[i] = 'Z' - diff;
    }
    else
```



```
        str[i] = new_code;
    }
} // for
printf("\nThe encrypted string is \n");
puts(str);
} // main
```

/*Program 7: Structure using array*/

```
#include <stdio.h>
#include <conio.h>
struct emp
{
    int empno;
    char name[10];
    int sal;
}e1[5];
void main()
{
    int i;
    clrscr();
    for(i=0;i < 5;i++)
    {
        printf("Enter the empno\n");
        scanf("%d",&e1[i].empno);
        printf("Enter the name\n");
        fflush();
        gets(e1[i].name);
        printf("Enter the salary\n");
```



```

scanf("%d",&e1[i].sal);
}
printf("The record is\n");
for(i=0;i < 5;i++)
{
printf("\n %4d",e1[i].empno);
printf("%8s",e1[i].name);
printf(" %4d\n",e1[i].sal);
}
getch();
}

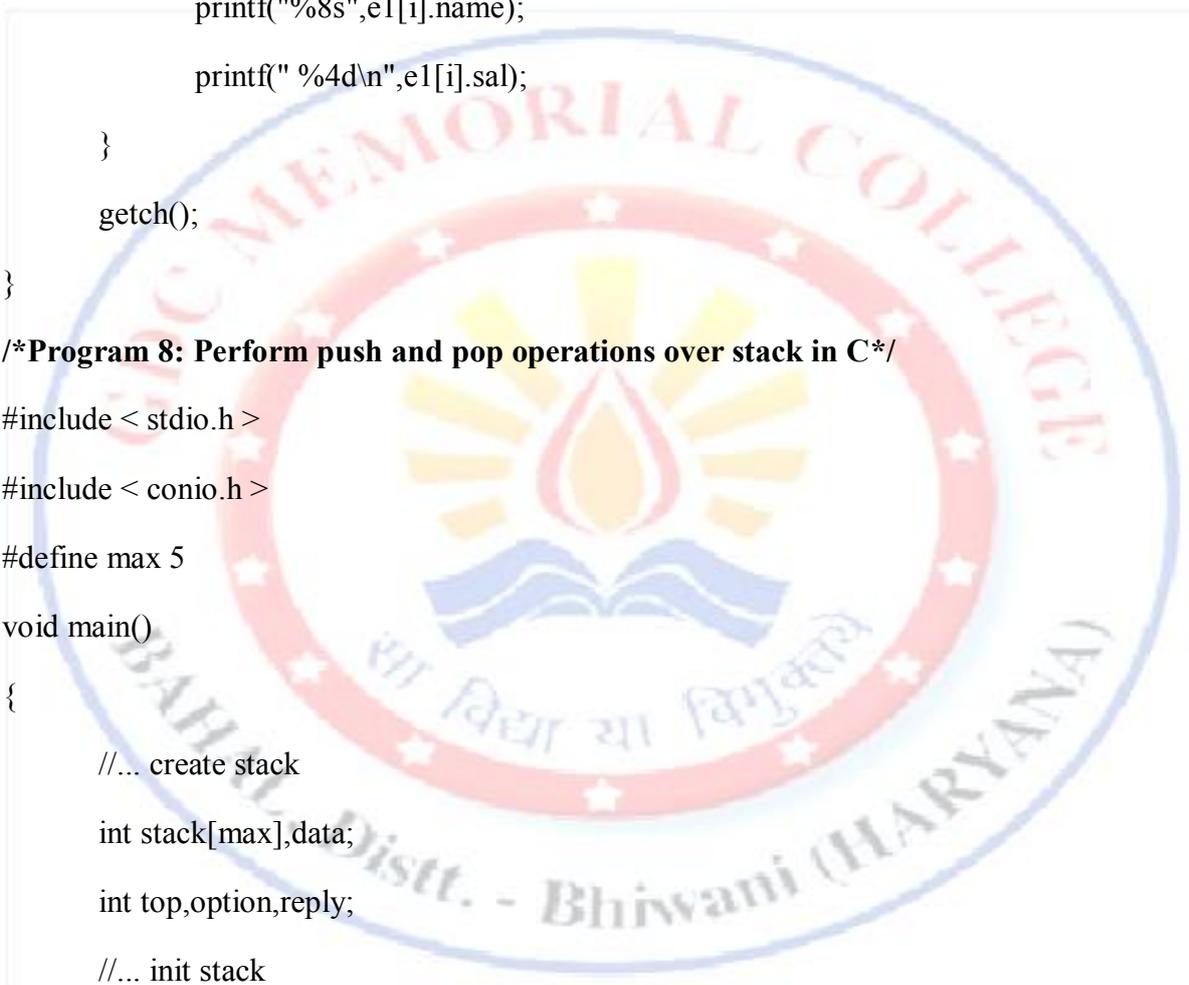
```

/*Program 8: Perform push and pop operations over stack in C*/

```

#include < stdio.h >
#include < conio.h >
#define max 5
void main()
{
//... create stack
int stack[max],data;
int top,option,reply;
//... init stack
top = -1;
clrscr();
do
{
printf("\n 1. push");
printf("\n 2. pop");

```



```

printf("\n 3. exit");

printf("\nSelect proper option : ");

scanf("%d",&option);

switch(option)
{

    case 1 : // push

        printf("\n Enter a value : ");

        scanf("%d",&data);

        reply = push(stack,&top,&data);

        if( reply == -1 )

            printf("\nStack is full");

        else

            printf("\n Pushed value");

        break;

    case 2 : // pop

        reply = pop ( stack,&top,&data);

        if( reply == -1 )

            printf("\nStack is empty");

        else

            printf("\n Popped value is %d",data);

        break;

    case 3 : exit(0);

} // switch

}while(1);

} // main

int push( int stack[max],int *top, int *data)

{

```

```

        if( *top == max -1 )
            return(-1);
        else
        {
            *top = *top + 1;
            stack[*top] = *data;
            return(1);
        } // else
    } // push
int pop( int stack[max], int *top, int *data)
{
    if( *top == -1 )
        return(-1);
    else
    {
        *data = stack[*top];
        *top = *top - 1;
        return(1);
    } //else
} // pop

```

/*Program 9: Sort numeric array using bubble sort*/

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

```
#include < conio.h >
```

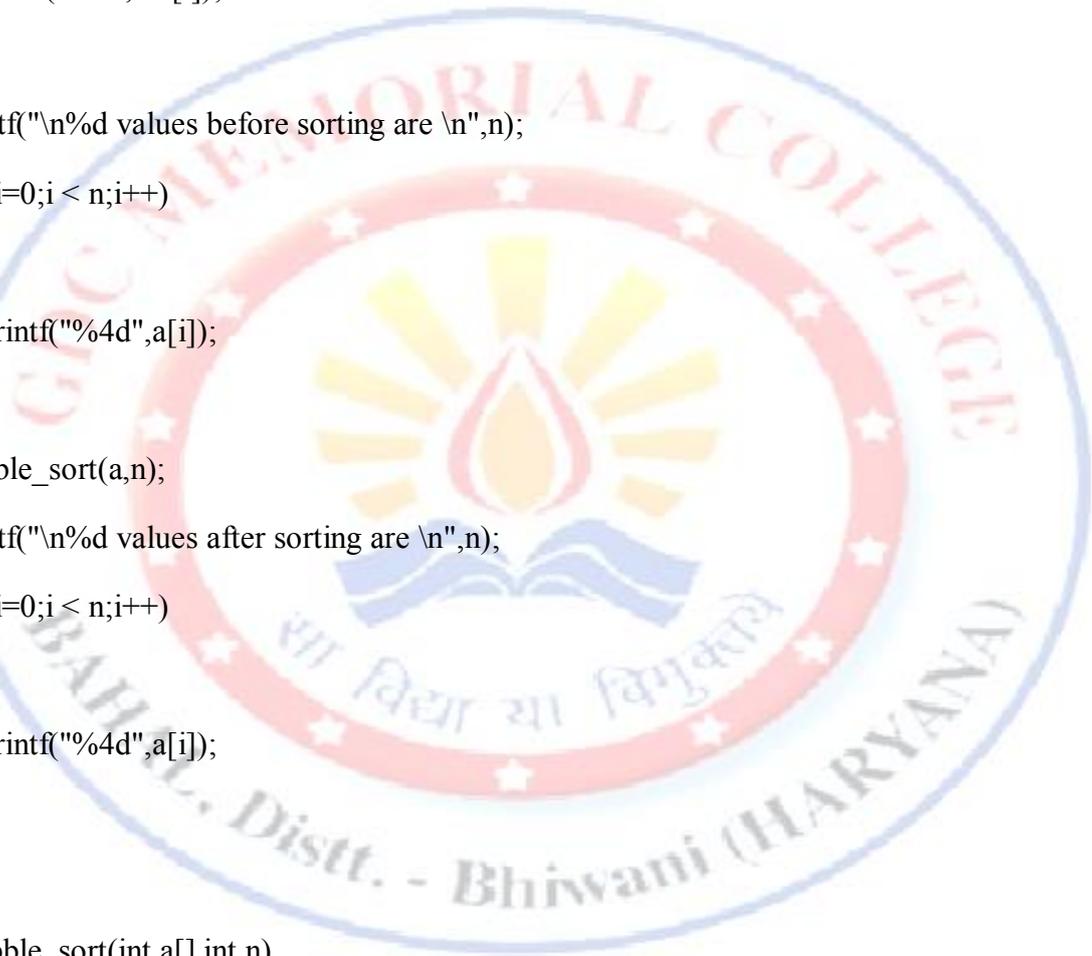
```
void main()
```

```
{
```

```
    int a[20],t;
```

```
    int i,j,n;
```

```
clrscr();
printf("\nEnter n :");
scanf("%d",&n);
printf("\nEnter %d values \n",n);
for(i=0;i < n;i++)
{
    scanf("%d",&a[i]);
}
printf("\n%d values before sorting are \n",n);
for(i=0;i < n;i++)
{
    printf("%4d",a[i]);
}
bubble_sort(a,n);
printf("\n%d values after sorting are \n",n);
for(i=0;i < n;i++)
{
    printf("%4d",a[i]);
}
}
int bubble_sort(int a[],int n)
{
    int i,j,k;
    int t;
    for(k=n;k>=1;k--)
    {
        for(i=0,j=1;j <=k;i++,j++)
```



```
{  
    if (a[j] < a[i])  
    {  
        t=a[i];  
        a[i]=a[j];  
        a[j]=t;  
    }  
}  
}  
return;  
}
```

/*Program 10: Sort n values using quick sort method*/

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

```
#include < conio.h >
```

```
void main()
```

```
{
```

```
int a[50];
```

```
int n;
```

```
clrscr();
```

```
printf("\nEnter n: ");
```

```
scanf("%d",&n);
```

```
read_data(a,n);
```

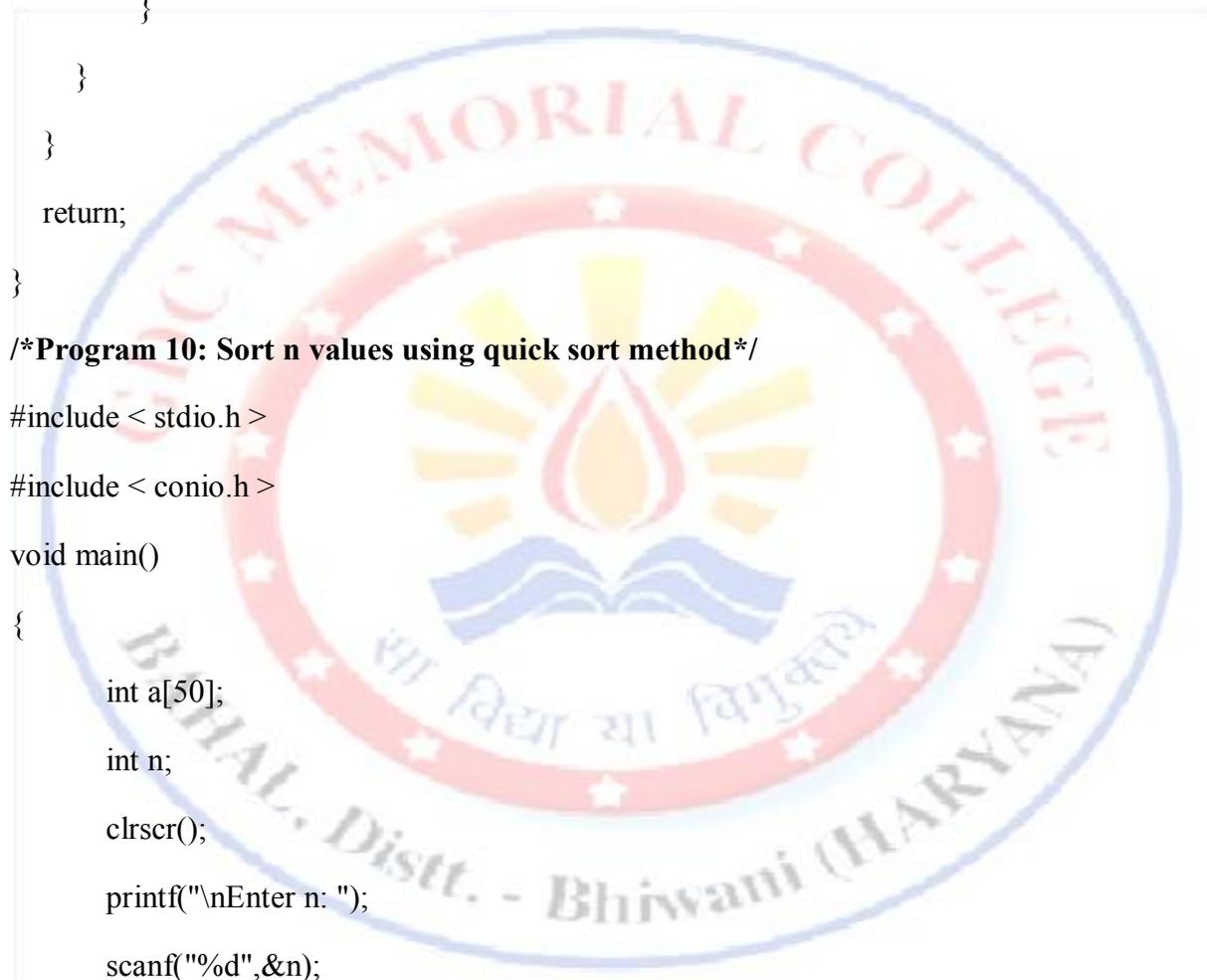
```
a[n]=9999;
```

```
printf("\nBefore sort :");
```

```
print_data(a,n);
```

```
qsort(a,0,n-1,n);
```

```
printf("\nAfter sort :");
```



```
    print_data(a,n);
}
int read_data(int a[],int max)
{
    int i;
    printf("\nEnter %d values \n",max);
    for(i=0; i < max; i++)
    {
        scanf("%d",&a[i]);
    }
    return;
}
```

```
int print_data(int a[],int max)
{
    int i;
    for(i=0; i < max; i++)
    {
        printf("%4d",a[i]);
    }
    return;
}
```

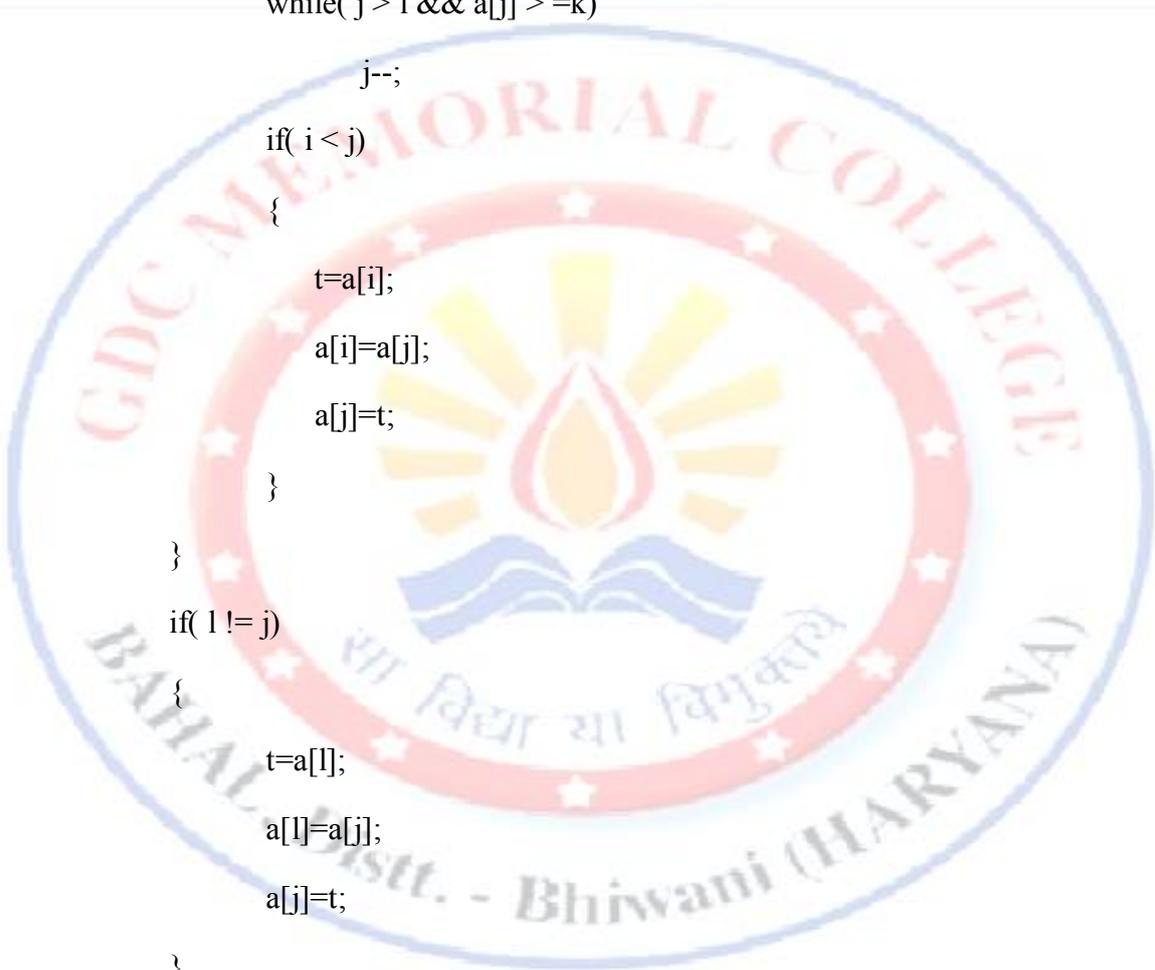
```
int quick_sort(int a[],int l,int h,int n)
{
    int i,j,k,t;
    if(l < h)
    {
        k=a[l];
```



```

i=l;
j=h+1;
while( i < j)
{
    while( i < h && a[i] <= k)
        i++;
    while( j > l && a[j] >=k)
        j--;
    if( i < j)
    {
        t=a[i];
        a[i]=a[j];
        a[j]=t;
    }
    if( l != j)
    {
        t=a[l];
        a[l]=a[j];
        a[j]=t;
    }
    printf("\nOutput :");
    print_data(a,n);
    qsort(a,l,j-1,n);
    qsort(a,j+1,h,n);
}
//print_data(a,n);

```



return;

}

