

NAME: **VENNELA G**

REG NO: **20BDS0146**

SUBJECT: **ADVANCED C  
PROGRAMMING**

LAB SLOT: **L47+48**

ASSESSMENT NO: **3**

**1. Write a C program to demonstrate the application of function pointer to avoid code redundancy. For example, sort a set of items of any type.**

**Note: You can use any sorting technique for sorting of set of items. Use the features of function pointer and void pointer.**

**CODE:**

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

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

```
int comp(const void *ptr1, const void *ptr2)
```

```
{
```

```
    if((* (int *)ptr1)==(* (int *)ptr2))
```

```
    {
```

```
        return 0;
```

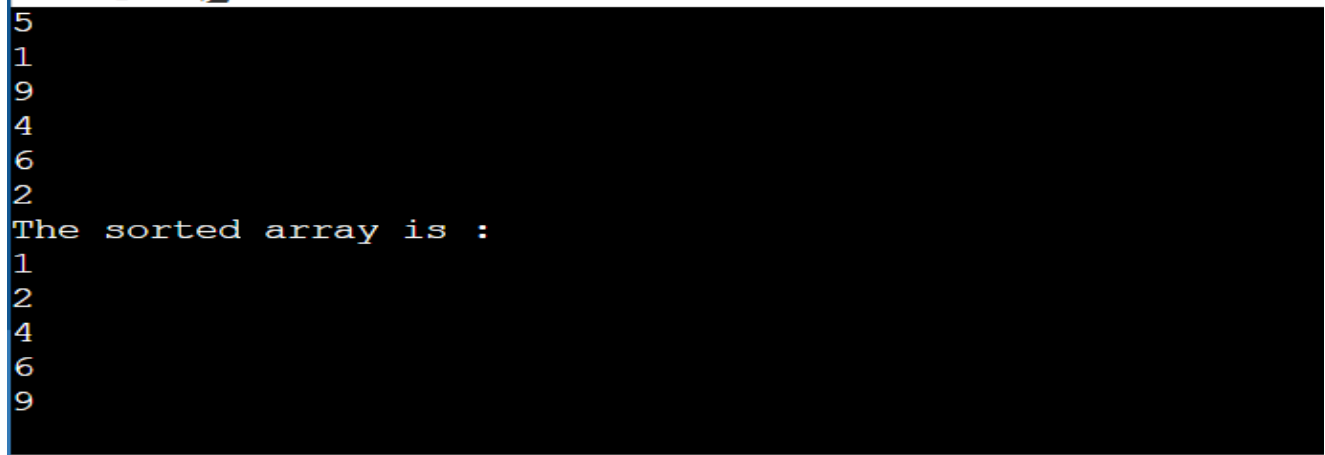
```
    }
```

```
    else if((* (int *)ptr1)<(* (int *)ptr2))
```

```
{  
    return -1;  
}  
  
else  
{  
    return 1;  
}  
}  
  
int main()  
{  
    int num;int arr[num];  
    scanf("%d",&num);  
    for(int i=0; i<num;i++)  
    {  
        scanf("%d",&arr[i]);  
    }  
  
    qsort((void*)arr,num,sizeof(arr[0]),comp);
```

```
printf("The sorted array is :\n");  
for(int i=0;i<num;i++)  
{  
    printf("%d\n",arr[i]);  
}  
}
```

**OUTPUT:**



```
5  
1  
9  
4  
6  
2  
The sorted array is :  
1  
2  
4  
6  
9
```

**2. Write a C program to demonstrate the application of function pointers as callback function.**

## CODE:

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

```
void add(int x, int y)
```

```
{printf("Addition Result is: %d\n",x+y);
```

```
}
```

```
void sub(int x1, int y1)
```

```
{printf("Subtraction Result is:%d\n",x1-y1);
```

```
}
```

```
void multi(int x2, int y2)
```

```
{printf("Multiplication Result is: %d\n",x2*y2);
```

```
}
```

```
void callback(void(*ptr1)(int,int))
```

```
{ptr1(15,5);
```

```
}
```

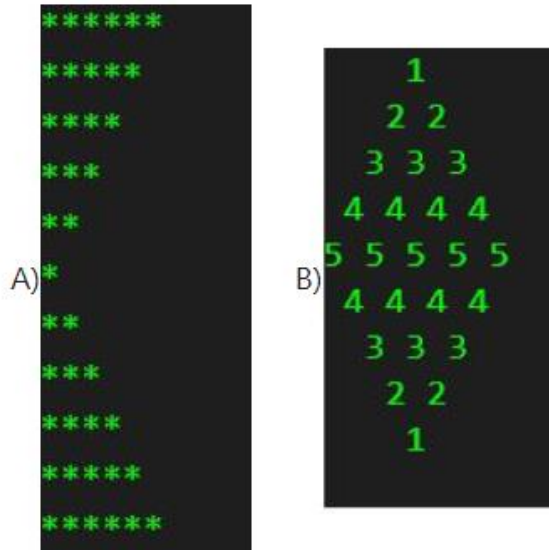
```
int main()
```

```
{  
    callback(add);  
    callback(sub);  
    callback(multi);  
    return 0;}
```

## OUTPUT:

```
Addition Result is: 20  
Subtraction Result is:10  
Multiplication Result is: 75  
  
...Program finished with exit code 0  
Press ENTER to exit console.█
```

### 3. Write a C program to print the following pattern:



#### CODE:

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

```
int main()
```

```
{int y1;int x1;
```

```
for(int i=0;i<6;i++)
```

```
{for(int j=6;j>i;j--)
```

```
{printf("*"); }
```

```
printf("\n");}
```

```
for(int i=0;i<5;i++)
```

```
{for(int j=-1;j<i+1;j++)  
{printf("*");}  
printf("\n");  
}
```

```
for(int i=0;i<6;i++)  
{x1=i;  
for(int j=5;j>i;j--)  
    {printf(" ");}  
for(int k=0;k<i;k++)  
    {printf("%d ",x1); }  
    printf("\n");}  
for(int i=0;i<5;i++)  
    {x1--;  
    for(int j=0;j<i+1;j++)  
        {printf(" ");}  
    for(int k=4;k>i;k--)  
        {printf("%d ",x1);}  
        printf("\n");  
    }
```



```
}
```

## OUTPUT:

```
*****
*****
****
***
**
*
**
***
****
*****
*****

  1
 2 2
3 3 3
4 4 4 4
5 5 5 5 5
4 4 4 4
 3 3 3
  2 2
   1

...Program finished with exit code 0
Press ENTER to exit console.□
```