USP Practice Assignment-3

1. WAP to find sum of two integer number using command line argument.

Ans-

```
#include <stdio.h>
int main(int argc, char *argv[])
{
int a,b,sum;
if(argc==3)
{
a = atoi(argv[1]);
b = atoi(argv[2]);
sum = a+b;
printf("Sum of %d, %d is: %d\n",a,b,sum);
return 0;
}
else
{
printf("Please use two values\n");
return -1;
}
}
```

2. WAP to check even and odd number using command line argument.

Ans-

```
#include <stdio.h>
int main(int argc, char *argv[])
```

```
{
int a;
if(argc==2)
{
a = atoi(argv[1]);
if(a%2==0)
printf("%d is Even\n",a);
else
printf("%d is Odd\n",a);
return 0;
}
else
{
printf("Please use one values\n");
return -1;
}
}
```

3. WAP to find factorial of a number using command line argument.

Ans-

```
#include <stdio.h>
int main(int argc, char *argv[])
{
int a;
if(argc==2)
{
a = atoi(argv[1]);
int fact=1;
for(int i=1;i<=a;i++)
{
fact = fact*i;
}
printf("The factorial of %d is: %d\n",a,fact);
return 0;
}
else
{
printf("Please use one values\n");
return -1;
}
}
```

4. WAP to check whether the number is palindrome or not.

Ans-

```
#include <stdio.h>
int main(int argc, char *argv[])
{
int a;
if(argc==2)
{
a = atoi(argv[1]);
int rem;
int num=0;
int n=a;
while(n!=0)
{
```

```
rem=n%10;
num = num*10+rem;
n=n/10;
}
if(num==a)
printf("The number %d is palindrome\n",a);
else
printf("The number %d is not palindrome\n",a);
return 0;
}
else
{
printf("Please use one values\n");
return -1;
}}
```

5. WAP to find largest number in an array using pointer.

Ans-

```
#include <stdio.h>
int main(int argc, char *argv[])
{
int arr[argc-1];
int size = argc-1;
for(int i=0;i<size;i++)
{
arr[i]=atoi(argv[i+1]);
printf("%d\n",arr[i]);
}
int *p;
p=&arr[0];
int max=arr[0];
for(int i=0;i<size;i++)
{
if(max<=*p)
max=*p;
p++;
}
printf("The maximum element is %d\n",max);
}
```

6. WAP to find max, min, average of 10 integers using array and pointer.

Ans-

```
#include <stdio.h>
int main(int argc, char *argv[])
{
int arr[argc-1];
int size = argc-1;
for(int i=0;i<size;i++)
{
arr[i]=atoi(argv[i+1]);
printf("%d\n",arr[i]);
}
int *p;
p=&arr[0];
int max=arr[0];
for(int i=0;i<size;i++)
{
if(max<=*p)
max=*p;
p++;
}
int *q;
q=&arr[0];
int min=arr[0];
for(int i=0;i<size;i++)
{
if(min>=*q)
max=*q;
q++;
}
int *r;
r=&arr[0];
int avg;
int sum=0;
for(int i=0;i<size;i++)
{
sum=sum+*r;
r++;
}
printf("The maximum element is %d\n",max);
printf("The minimum element is %d\n",min);
printf("The average is %d\n",(sum/size-1));
}
```

7. WAP to access element of an array using pointer.

Ans-

```
#include <stdio.h>
int main(int argc, char *argv[])
{
int arr[argc-1];
int size = argc-1;
for(int i=0;i<size;i++)
{
arr[i]=atoi(argv[i+1]);
}
int *p;
p=&arr[0];
for(int i=0;i<size;i++)
{
printf("%d\n",*p);
p++;
}
}
```

8. WAP to add two matrix using multidimensional array.

Ans-

```
#include <stdio.h>
int main(int argc, char *argv[])
{
int arr1[2][2];
int arr2[2][2];
int result [2][2];
int size = argc-1;

arr1[0][0]=atoi(argv[1]);
arr1[0][1]=atoi(argv[2]);
arr1[1][0]=atoi(argv[3]);
arr1[1][1]=atoi(argv[4]);


arr2[0][0]=atoi(argv[5]);
arr2[0][1]=atoi(argv[6]);
arr2[1][0]=atoi(argv[7]);
arr2[1][1]=atoi(argv[8]);

result[0][0]=arr1[0][0]+arr2[0][0];
result[0][1]=arr1[0][1]+arr2[0][1];
result[1][0]=arr1[1][0]+arr2[1][0];
result[1][1]=arr1[1][1]+arr2[1][1];
int i,j;
for(i=0;i<2;i++)
{
  for(j=0;j<2;j++)
  {
   printf("%d ",result[i][j]);
  }
    printf("%c ",'\n');
}}
```

SORTING

Binary Search-

```
#include<stdio.h>
int bsearch(int A[ ], int lo, int hi, int key);
int
main()
{
        int n;
        printf("Enter the size of the array => ");
        scanf("%d",&n);
        int Arr[n];
        printf("Enter array elements in sorted order => \n");
        int i;
        for(i=0;i<n;i++)
        {
                printf("Enter Array Element-%d => ",i+1);
                scanf("%d",&Arr[i]);
        }
         int k;
         printf("Enter the element you want to search => ");
         scanf("%d",&k);
         int ind=bsearch(Arr,0,n-1,k);
         printf("The index of the element %d is %d",k,ind);
         printf("\n");

}
int bsearch(int A[ ], int lo, int hi, int key)
```

```
{
    int mid;
    if(lo<hi)
    {
        int mid=(lo+hi)/2;
       if(mid==key)
            return mid;
        else if(key<mid)
            return bsearch(A,lo,mid-1,key);
        else
            return bsearch(A,mid+1,hi,key);
    }
}
```

<u>Bubble Sort</u>-

```
#include <stdio.h>
void bubble(int arr[ ], int size);
int main(int argc, char *argv[ ])
{
 int arr[argc-1];
 int size = argc-1;
    for(int i=0;i<size;i++)
    {
        arr[i]=atoi(argv[i+1]);

    }

    bubble(arr,size);
```

```
        printf("SORTED ARRAY");
        for(int i=0;i<size;i++)
        {
                printf("%d ", arr[i]);
        }



}

void bubble(int a[ ], int size)
{
 int i,j,temp;
  for (i=0;i<size-1;i++)
   for (j=i+1;j<size;j++)
    {
         if (a[j]<a[i])
               {
                        temp=a[i];
                        a[i]=a[j];
                        a[j]=temp;
                }

    }
}
```

Insertion Sort-

```
#include <stdio.h>
void insertion(int arr[ ], int size);
int main(int argc, char *argv[ ])
{
 int arr[argc-1];
 int size = argc-1;
         for(int i=0;i<size;i++)
         {
                  arr[i]=atoi(argv[i+1]);

         }

         insertion(arr,size);
         printf("SORTED ARRAY");
         for(int i=0;i<size;i++)
         {
                   printf("%d ", arr[i]);
         }



}

void insertion(int arr[ ], int size)
{
   int i, key, j;
   for (i = 1; i < size; i++)
```

```
  {
    key = arr[i];
    j = i-1;

    while (j >= 0 && arr[j] > key)
    {
      arr[j+1] = arr[j];
      j = j-1;
    }
    arr[j+1] = key;
  }
}
```

Selection Sort-

```
#include <stdio.h>
void selection(int arr[ ], int size);
int main(int argc, char *argv[ ])
{
 int arr[argc-1];
 int size = argc-1;
     for(int i=0;i<size;i++)
     {
          arr[i]=atoi(argv[i+1]);

     }

     selection(arr,size);
     printf("SORTED ARRAY");
```

```
         for(int i=0;i<size;i++)
         {
                    printf("%d ", arr[i]);
         }
}
void selection(int arr[ ], int n)
{
    int i, j, min_idx, temp;


    for (i = 0; i < n-1; i++)
    {

        min_idx = i;
        for (j = i+1; j < n; j++)
          if (arr[j] < arr[min_idx])
            min_idx = j;


         temp=arr[min_idx];
         arr[min_idx]=arr[i];
         arr[i]=temp;

    }
}
```