NAME: AGNIV PRAMANICK **SECTION:** A **USN:** 1NT21IS017 **DATE:** 29/12/22

Q. Write a program to find the position of the element using binary search.

THEORY

Binary search is a searching algorithm for finding an element’s position in a sorted array. In this approach, the element is always searched in the middle of a portion of an array.

Binary search can be implemented only on a sorted list of items. If the elements are not sorted already, we need to sort them first.

Binary Search Algorithm can be implemented in two ways,

1. Iterative Method
2. Recursive Method

ALGORITHM

Step1: select the element to be searched

Step2: set two pointers for low and high at the lowest and the highest position.

Step3: Find the middle element of array using, middle= (initial_value + end_value) / 2;

Step4: If middle = element, return middle. Else, compare element to be searched with mid element.

Step5: if middle > element, call the function with end_value = middle - 1.

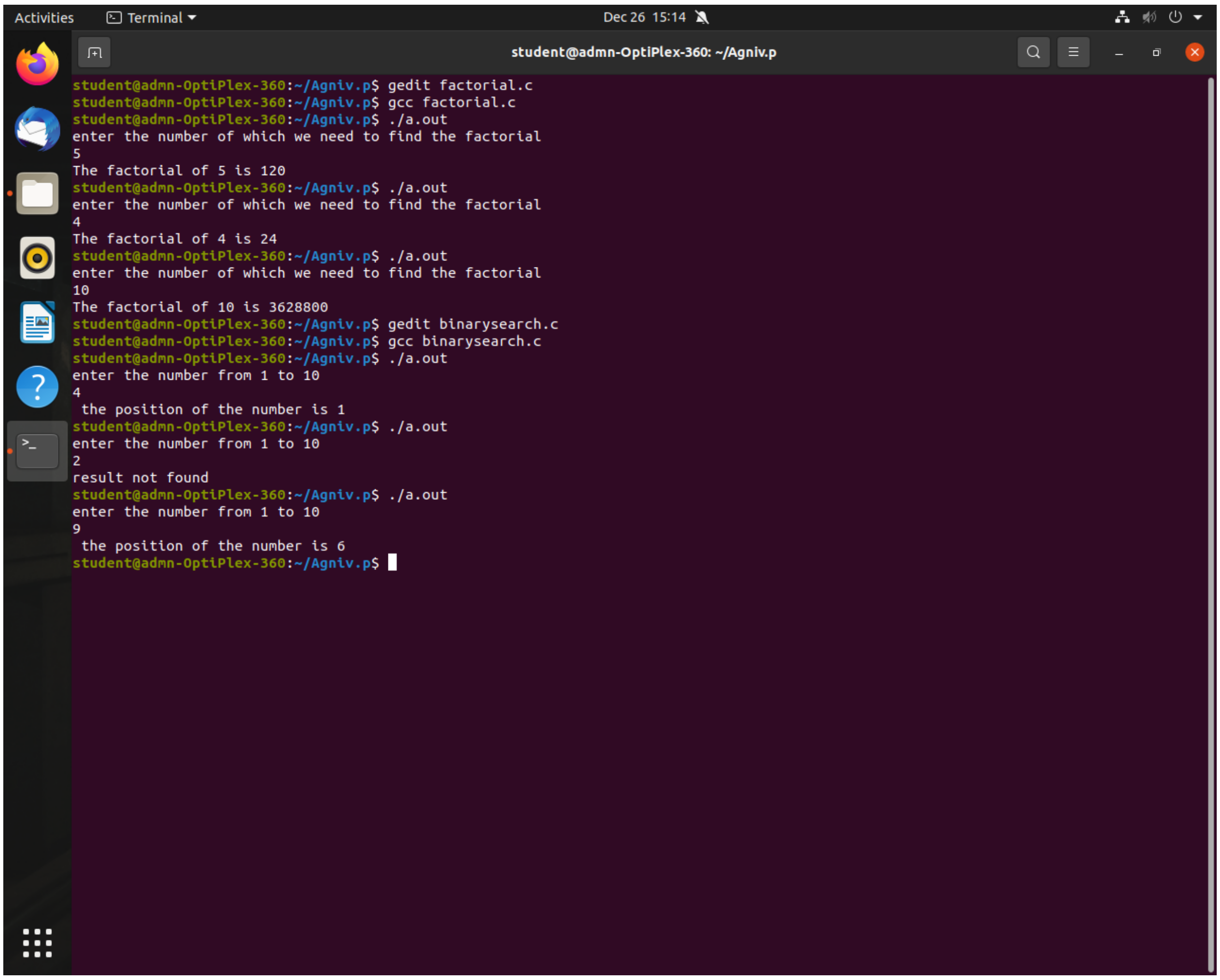
Step6: if middle < element, call the function with start_value = middle + 1.

Step7: exit.

CODE

```
#include<stdio.h>
#include<stdlib.h>       //header files are included
#include<string.h>

 //we define our user defined function which will be used to get the tower of hanoi order

int bins(int array[],int x,int high,int low)
{
  if(high>=low)  // if value of high is more than low
  {
   int mid=(low+high)/2;  //to find the middle value, add both high and low then divide by 2
   {
    if(array[mid]==x) //if the number chosen by user is equal to mid value, return mid
    {
      return mid;
    }
    else if(array[mid]>x) //number is less than the mid value then reduce the array from low to mid
    {
      return bins(array,x,mid-1,low); //the value of high will be mid-1 , function is called
    }
    else            //if the number is more than the mid value then reduce the array from mid to high
    {
      return bins(array,x,high,mid+1);  //the value of low will be mid+1 ,function is called
    }
   }
  }
  else if(low>high)  //if in case low value is more than high, return -1
  {
   return -1;
  }
}

//now, the main function starts from here

int main(void)
{
  int x;
  int array[]={3,4,5,6,7,8,9,10};          //the elements are already introduced in the array
  int n=sizeof(array)/sizeof(array[0]);     // divide the array size by array size at index 0
  printf("enter the number from 1 to 10\n");
  scanf("%d",&x);                          //user will select elements from 1 to 10
  int f=bins(array,x,n-1,0);                 //call the function
  if (f==-1)
  {
    printf("result not found \n");     // if return value is -1
  }
  else
  {
    printf(" the position of the number is %d \n",f);    //output to the user is given
  }
}
//END OF PROGRAM
```

OUTPUT SCREENSHOT



```
student@admn-OptiPlex-360:~/Agniv.p$ gedit factorial.c
student@admn-OptiPlex-360:~/Agniv.p$ gcc factorial.c
student@admn-OptiPlex-360:~/Agniv.p$ ./a.out
enter the number of which we need to find the factorial
5
The factorial of 5 is 120
student@admn-OptiPlex-360:~/Agniv.p$ ./a.out
enter the number of which we need to find the factorial
4
The factorial of 4 is 24
student@admn-OptiPlex-360:~/Agniv.p$ ./a.out
enter the number of which we need to find the factorial
10
The factorial of 10 is 3628800
student@admn-OptiPlex-360:~/Agniv.p$ gedit binarysearch.c
student@admn-OptiPlex-360:~/Agniv.p$ gcc binarysearch.c
student@admn-OptiPlex-360:~/Agniv.p$ ./a.out
enter the number from 1 to 10
4
 the position of the number is 1
student@admn-OptiPlex-360:~/Agniv.p$ ./a.out
enter the number from 1 to 10
2
result not found
student@admn-OptiPlex-360:~/Agniv.p$ ./a.out
enter the number from 1 to 10
9
 the position of the number is 6
student@admn-OptiPlex-360:~/Agniv.p$
```