POLYNIMIAL EQUATION SOLUTION:-


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

struct myarray{
 int No_of_elements;
 int *ptr;
};

int createarray(struct myarray *a,int a_size){
 a->No_of_elements=a_size;
 a->ptr=(int*)malloc(a_size*sizeof(int));
}
int inputpos(int d,struct myarray * arr,struct myarray * duparr,int p,int o){
 int b;
 if(p==1){
  int x=1;
  for(int i=0;i<d;i++){
   printf("\nEnter the cofficient of %dth term in equation from right side: \n",i+1);
   int q;

   scanf("%d",&q);
   (arr->ptr)[i]=q;
   (arr->ptr)[i]=pow(x,i)*((arr->ptr)[i]);
   //printf("\t (arr->ptr)[i] after %d\n",(arr->ptr)[i]);
  }
  //printf("\n\n%d\n",x);
  //-----------------
         for(int i=d-1;i>=0;i--){
          if(i==0){
              //printf("(%d)",(arr->ptr)[i]);
              break;
          }
          //printf("(%d)+",(arr->ptr)[i]);
         }
      //-----------------
  int count=0;
  for(int i=0;i<d;i++){
        //printf("\narri is (%d) and count is %d",(arr->ptr)[i],count);
         count=count+(arr->ptr)[i];
        }
        printf("\nValue of equation for x as %d is: %d",x,count);
        b=count;
 }
 else{
  int x=2,gog=0;
  while(x<o){
   //printf("\n\n%d\n",x);
      for(int i=0;i<d;i++){
```

```
    //printf("\t (arr->ptr)[i] before %d\n",(arr->ptr)[i]);
      (duparr->ptr)[i]=pow(x,i)*((arr->ptr)[i]);
    //printf("\t (duparr->ptr)[i] after %d\n",(duparr->ptr)[i]);
      }
   //-----------------
        for(int i=d-1;i>=0;i--){
         if(i==0){
            //printf("(%d)",(duparr->ptr)[i]);
            break;
         }
         //printf("(%d)+",(duparr->ptr)[i]);
        }
     //-----------------
     int count=0,countpre=b;
        for(int i=0;i<d;i++){
         //printf("\narri is (%d) and count is %d",(duparr->ptr)[i],count);
         count=count+(duparr->ptr)[i];
        }
        printf("\nValue of equation for x as %d is: %d",x,count);
        if((count>0&&countpre<0)||(countpre>0&&count<0)||(count==0||countpre==0)||(count==0&&countpre==0)){
         return x;
   }
   countpre=count;
     x++;
     }
    if(gog==0){
     return 0;
  }
 }
}
int inputneg(int d,struct myarray * arr,struct myarray * duparr,int p,int o){
 int b;
 if(p==1){
  int x=-1;
  for(int i=0;i<d;i++){
   printf("\nEnter the cofficient of %dth term in equation from right side: \n",i+1);
   int q;
   scanf("%d",&q);
   (arr->ptr)[i]=q;
   (arr->ptr)[i]=pow(x,i)*((arr->ptr)[i]);
   //printf("\t (arr->ptr)[i] after %d\n",(arr->ptr)[i]);
  }
  //printf("\n\n%d\n",x);
  //-----------------
        for(int i=d-1;i>=0;i--){
         if(i==0){
            //printf("(%d)",(arr->ptr)[i]);
            break;
         }
         //printf("(%d)+",(arr->ptr)[i]);
        }
  //-----------------
  int count=0;
  for(int i=0;i<d;i++){
        //printf("\narri is (%d) and count is %d",(arr->ptr)[i],count);
```

```
         count=count+(arr->ptr)[i];
        }
        printf("\nValue of equation for x as %d is: %d",x,count);
        b=count;
 }
 else{
  int x=-2,gog=0;
  while(x>(-o)){
   //printf("\n\n%d\n",x);
      for(int i=0;i<d;i++){
       if(i==0){
        (duparr->ptr)[i]=(arr->ptr)[i];
        continue;
    }
    //printf("\t (arr->ptr)[i] before %d\n",(arr->ptr)[i]);
       (duparr->ptr)[i]=pow(x,i)*((arr->ptr)[i]);
       (duparr->ptr)[i]=-(duparr->ptr)[i];
    //printf("\t (duparr->ptr)[i] after %d\n",(duparr->ptr)[i]);
      }
   //-----------------
         for(int i=d-1;i>=0;i--){
          if(i==0){
              //printf("(%d)",(duparr->ptr)[i]);
              break;
          }
          //printf("(%d)+",(duparr->ptr)[i]);
         }
      //-----------------
      int count=0,countpre=b;
         for(int i=0;i<d;i++){
          //printf("\narri is (%d) and count is %d",(duparr->ptr)[i],count);
          count=count+(duparr->ptr)[i];
         }
         printf("\nValue of equation for x as -(%d) is: %d",x,count);
         if((count>0&&countpre<0)||(countpre>0&&count<0)||(count==0||countpre==0)||(count==0&&countpre==0)){
          return x;
   }
   countpre=count;
      x--;
     }
     if(gog==0){
      return 0;
  }
 }
}

int main()
{
 //  /*testing  V

 int x;
 printf("For how many integers you want to calculate equation value: \n");
 scanf("%d",&x);
 int d;
 struct myarray arr;
 struct myarray duparr;
 printf("Enter the degree of equation: \n");
 scanf("%d",&d);
 int neg,pos;
 createarray(&arr,d+1);
 createarray(&duparr,d+1);
 int value1,value2,pos1,pos2,neg1,neg2;
 printf("\n\n\n\t\t\t-: USING POSITIVE INTEGERS :-\n");
 inputpos(d+1,&arr,&duparr,1,x+1);
 value1=inputpos(d+1,&arr,&duparr,0,x+1);
 if(value1==0){
        printf("\n\nZero not found on and between any of the integers\n-------------\n");
 }
 else{
  pos1=value1;
  neg1=pos1-1;
 printf("\n\nZero lies between both or on any one of these values of X: %d, %d\n-------------\n",neg1,pos1);
 }
 printf("\n\n\n\t\t\t-: USING NEGETIVE INTEGERS :-\n");
 inputneg(d+1,&arr,&duparr,1,x+1);
 value2=inputpos(d+1,&arr,&duparr,0,x+1);
 if(value2==0){
        printf("\n\nZero not found on and between any of the integers\n-------------n");
 }
 else{
  pos2=value2;
  neg2=pos2-1;
 printf("\n\nZero lies between both or on any one of these values of X: %d, %d\n-------------\n",neg2,pos2);
 }

 //  testing  ^ */
}
```


Designed by: Jatin Sharma