20MAI1001
PRABHU GANTAYAT

ALGO LAB 2

Lab 2 Arrays – 13/08/2020

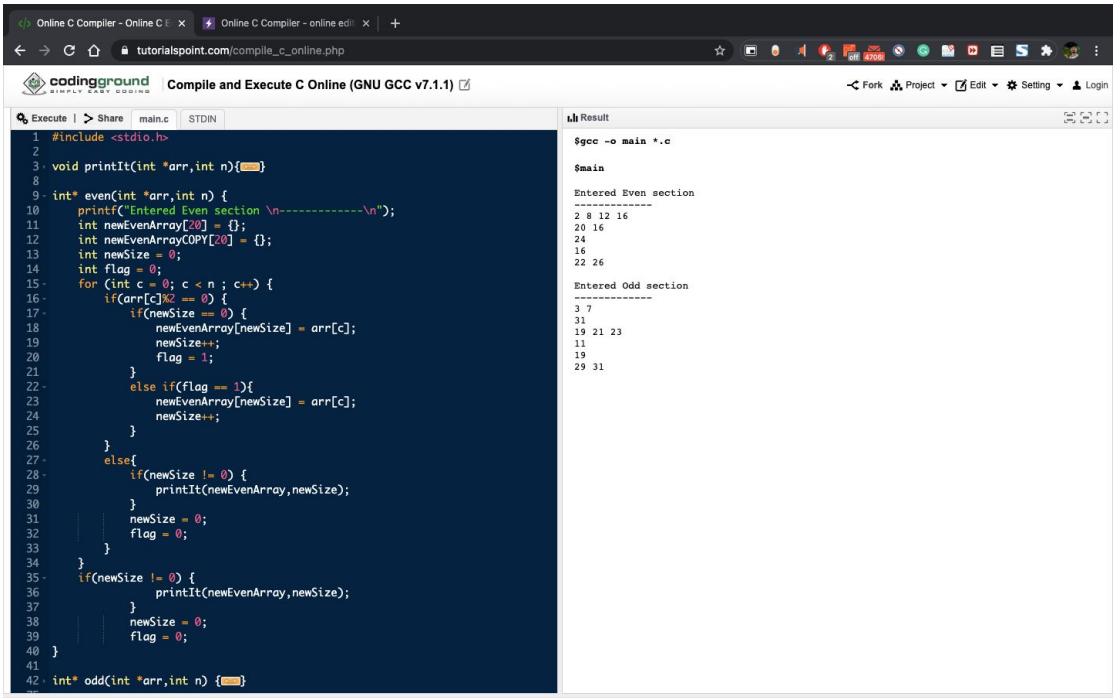
1) Assume A is an array of size n. A group S = { Y_k €A : 1<=k<= m } is said be a " group of even numbers" of size m , if all elements in the group are consecutive elements in the array A and each number is divisible by 2 .Otherwise, it is a "group of odd numbers" i.e., all elements in the group are consecutive elements in the array A and each number is not divisible by 2.

Ans.

```
#include <stdio.h>

void printIt(int *arr,int n){
  for (int c = 0; c < n ; c++)
      printf("%d ", arr[c]);
   printf("\n");
}

int* even(int *arr,int n) {
   printf("Entered Even section \n-------------\n");
   int newEvenArray[20] = {};
   int newEvenArrayCOPY[20] = {};
   int newSize = 0;
   int flag = 0;
         for (int c = 0; c < n ; c++) {
            if(arr[c]%2 == 0) {
               if(newSize == 0) {
                  newEvenArray[newSize] = arr[c];
                  newSize++;
                  flag = 1;
               }
               else if(flag == 1){
                  newEvenArray[newSize] = arr[c];
                  newSize++;
               }
            }
            else{
               if(newSize != 0) {
                  printIt(newEvenArray,newSize);
               }
         newSize = 0;
         flag = 0;
            }
         }
   if(newSize != 0) {
                  printIt(newEvenArray,newSize);
```

```
                }
            newSize = 0;
            flag = 0;
}

int* odd(int *arr,int n) {
    printf("\nEntered Odd section \n-------------\n");
            int newEvenArray[20] = {};
    int newEvenArrayCOPY[20] = {};
    int newSize = 0;
    int flag = 0;
            for (int c = 0; c < n ; c++) {
                if(arr[c]%2 != 0) {
                    //printf("%d, ",arr[c]);
                    if(newSize == 0 && flag == 0) {
                        newEvenArray[newSize] = arr[c];
                        newSize++;
                        flag = 1;
                    }
                    else if(flag == 1){
                        newEvenArray[newSize] = arr[c];
                        newSize++;
                    }
                }
                else{
                    if(newSize != 0) {
                        printIt(newEvenArray,newSize);
                    }
            newSize = 0;
            flag = 0;
                }
            }
            if(newSize != 0) {
                        printIt(newEvenArray,newSize);
                    }
            newSize = 0;
            flag = 0;
}

int main() {
    int newArr[100] = { 0 };
            int arr1[]={3,7,2,8,12,16,31,20,16,19,21,23,24,11,16,19,22,26,29,31};
            int* arr = arr1;
            int* arr2 = arr1;
            int n = sizeof(arr1)/sizeof(arr1[0]);
            int newPos = 0;
```

```
        int dup_n = n;

        even(arr,n);
        odd(arr2,n);

        //printf("%d",n);
    return 0;
}
```



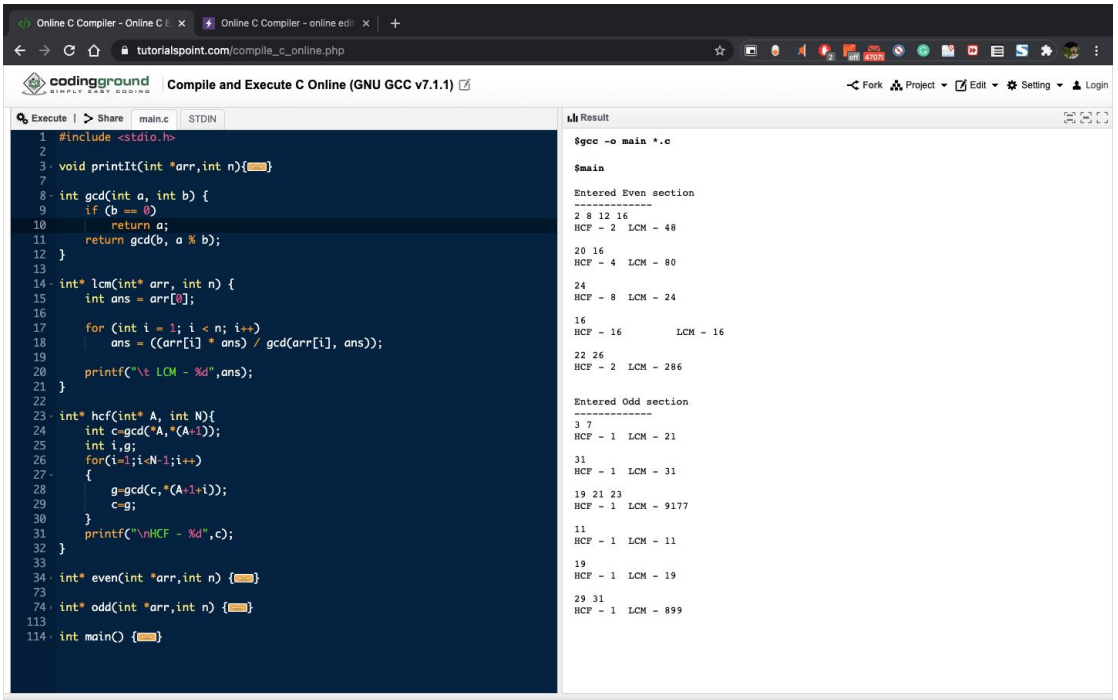
2) Write a program to identify the even groups and odd groups in the given array and calculate the LCM (Least Common Multiple) of each group.

```
#include <stdio.h>

void printIt(int *arr,int n){
  for (int c = 0; c < n ; c++)
      printf("%d ", arr[c]);
}

int gcd(int a, int b) {
    if (b == 0)
        return a;
    return gcd(b, a % b);
}

int* lcm(int* arr, int n) {
```

```
    int ans = arr[0];

    for (int i = 1; i < n; i++)
        ans = ((arr[i] * ans) / gcd(arr[i], ans));

    printf("\t LCM - %d",ans);
}

int* hcf(int* A, int N){
    int c=gcd(*A,*(A+1));
    int i,g;
    for(i=1;i<N-1;i++)
    {
        g=gcd(c,*(A+1+i));
        c=g;
    }
    printf("\nHCF - %d",c);
}

int* even(int *arr,int n) {
    printf("Entered Even section \n-------------\n");
    int newEvenArray[20] = {};
    int newEvenArrayCOPY[20] = {};
    int newSize = 0;
    int flag = 0;
            for (int c = 0; c < n ; c++) {
                if(arr[c]%2 == 0) {
                    if(newSize == 0) {
                        newEvenArray[newSize] = arr[c];
                        newSize++;
                        flag = 1;
                    }
                    else if(flag == 1){
                        newEvenArray[newSize] = arr[c];
                        newSize++;
                    }
                }
                else{
                    if(newSize != 0) {
                        printIt(newEvenArray,newSize);
                        hcf(newEvenArray,newSize);
                        lcm(newEvenArray,newSize);
                        printf("\n\n");

                    }
            newSize = 0;
```

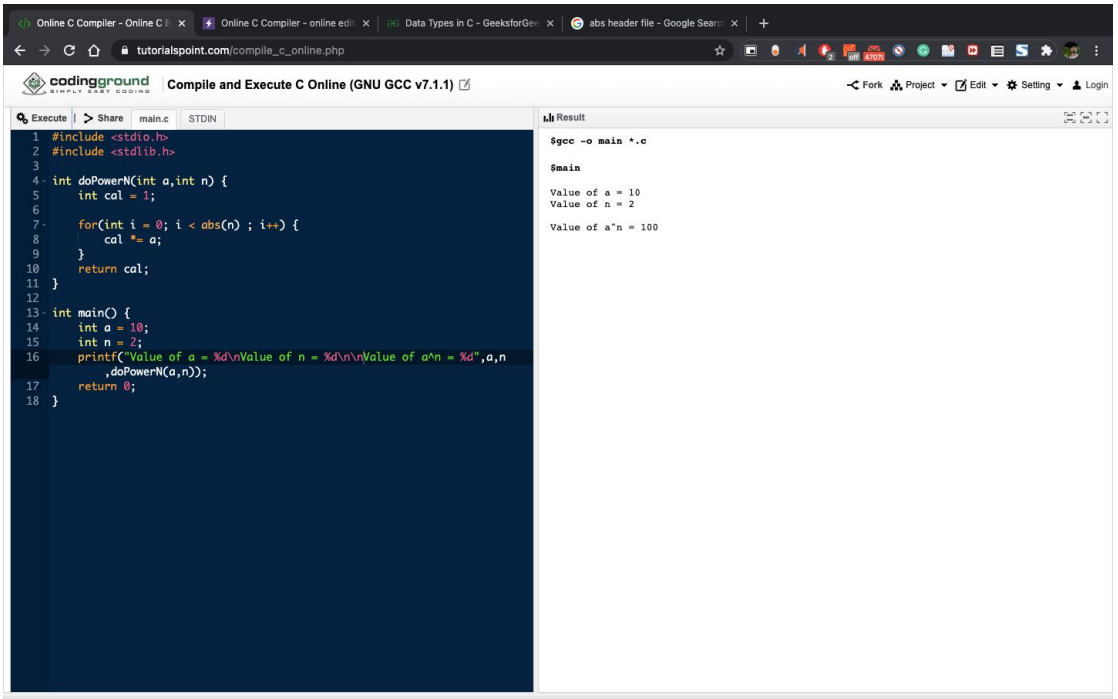
```
            flag = 0;
              }
            }
    if(newSize != 0) {
                    printIt(newEvenArray,newSize);
                    hcf(newEvenArray,newSize);
                    lcm(newEvenArray,newSize);
                    printf("\n\n");
                }
            newSize = 0;
            flag = 0;
}

int* odd(int *arr,int n) {
    printf("\nEntered Odd section \n-------------\n");
            int newEvenArray[20] = {};
    int newEvenArrayCOPY[20] = {};
    int newSize = 0;
    int flag = 0;
            for (int c = 0; c < n ; c++) {
              if(arr[c]%2 != 0) {
                //printf("%d, ",arr[c]);
                if(newSize == 0 && flag == 0) {
                    newEvenArray[newSize] = arr[c];
                    newSize++;
                    flag = 1;
                }
                else if(flag == 1){
                    newEvenArray[newSize] = arr[c];
                    newSize++;
                }
              }
              else{
                if(newSize != 0) {
                    printIt(newEvenArray,newSize);
                    hcf(newEvenArray,newSize);
                    lcm(newEvenArray,newSize);
                    printf("\n\n");
                }
            newSize = 0;
            flag = 0;
              }
            }
            if(newSize != 0) {
                    printIt(newEvenArray,newSize);
                    hcf(newEvenArray,newSize);
```

```
                lcm(newEvenArray,newSize);
                printf("\n\n");
            }
        newSize = 0;
        flag = 0;
}

int main() {
    int newArr[100] = { 0 };
        int arr1[]={3,7,2,8,12,16,31,20,16,19,21,23,24,11,16,19,22,26,29,31};
        int* arr = arr1;
        int* arr2 = arr1;
        int n = sizeof(arr1)/sizeof(arr1[0]);
        int newPos = 0;
        int dup_n = n;

        even(arr,n);
        odd(arr2,n);

        //printf("%d",n);
    return 0;
}
```



3) Let be an integer and be a nonnegative integer. Compute . In other words, we ask for a program that does not change the values of and but assigns the vaue of to another variable say .

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

int doPowerN(int a,int n) {
   int cal = 1;

   for(int i = 0; i < abs(n) ; i++) {
      cal *= a;
   }
   return cal;
}

int main() {
   int a = 10;
   int n = 2;
   printf("Value of a = %d\nValue of n = %d\n\nValue of a^n =
%d",a,n,doPowerN(a,n));
   return 0;
}
```



4) Let A[1…n] be an integer array of size n. Assume that the elements of the array are not in sorted order. Our goal is to re-arrange the elements in sorted order (increasing order of values). We will try to achieve that by using the following logic: We define an operation called SWAP(1,j) where the jth element of the array is swapped with the element located in the 1st position of the array A, i.e. A[1]. Use this swap operation as many times as you need to swap any element of the array with the element in the 1st position. After a set of swap operations, the elements of the array should be re-arranged in sorted order. Write a program that will achieve this goal.

```
#include <stdio.h>
int* swap(int *arr,int pos)
{
        int head = arr[0];
        arr[0] = arr[pos];
        arr[pos] = head;
        return arr;
}

void printIt(int *arr,int n){
  for (int c = 0; c < n ; c++)
      printf("%d ", arr[c]);
}

int* crop(int* arr){
   arr = arr;
   return arr;
}

int main()
{
   // 20MAI1001 PRABHU
   int newArr[100] = { 0 };
        int arr1[]={8,53,87,23,6,3,10,20,30,40};
        int* arr = arr1;
        int n = sizeof(arr1)/sizeof(arr1[0]);
        int newPos = 0;
        int dup_n = n;
   int i,j;
        for (i = 0; i < n; i++) {
           int head = *arr;
           printf("%d'th iteration - ",i+1);
           printf("dep_n = %d\n",dup_n);
           for (j = 1; j < dup_n; j++) {
              if(*(arr+j) > head){
```

```
                head = *(arr+j);
                swap(arr,j);
                newPos = j;
            }
        }
        swap(arr,dup_n-1);
        dup_n--;
        printf("%d is highest | Input array - ",head);
        printIt(arr,n);
        printf("\n\n");
    }

  printf("Final non-decreasing array - ");
  printIt(arr,n);
      return 0;
}
```