**Randomized Algorithms(Las vegas and Monte Carlo)**

**Randomized Quick sort algorithm**

**Algorithm for random pivoting using Hoare Partitioning**

partition(arr[], lo, hi)

pivot = arr[lo]

i = lo - 1 // Initialize left index

j = hi + 1 // Initialize right index

while(True)

// Find a value in left side greater than pivot

do

i = i + 1

while arr[i] < pivot

// Find a value in right side smaller than pivot

do

j = j - 1

while arr[j] > pivot

if i >= j then

return j

else

swap arr[i] with arr[j]

end while

partition\_r(arr[], lo, hi)

r = Random number from lo to hi

Swap arr[r] and arr[lo]

return partition(arr, lo, hi)

quicksort(arr[], lo, hi)

if lo < hi

p = partition\_r(arr, lo, hi)

quicksort(arr, lo, p)

quicksort(arr, p+1, hi)

**Write program to implement to Las Vegas Algorithm**

**Randomized Quick Sort**

**SOURCE CODE:**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <time.h>**

**int n;**

**void random\_shuffle(int arr[])**

**{**

**srand(time(NULL));**

**int i, j, temp;**

**for (i=n-1;i>0; i--)**

**{**

**j = rand()%(i + 1);**

**temp = arr[i];**

**arr[i] = arr[j];**

**arr[j] = temp;**

**}**

**}**

**void swap(int \*a, int \*b)**

**{**

**int temp;**

**temp = \*a;**

**\*a = \*b;**

**\*b = temp;**

**}**

**int partion(int arr[],int low,int high)**

**{**

**//p low r-high**

**int pivotIndex=low+rand()%(high-low+1); //generates a random number as a pivot**

**int pivot,j;**

**int i=low-1;**

**pivot=arr[pivotIndex];**

**swap(&arr[pivotIndex], &arr[high]);**

**for (j=low;j<high;j++)**

**{**

**if (arr[j]<pivot)**

**{**

**i++;**

**swap(&arr[i], &arr[j]);**

**}**

**}**

**swap(&arr[i+1], &arr[high]);**

**return i + 1;**

**}**

**void quick\_sort(int arr[], int low, int high)**

**{**

**int j;**

**if (low<high)**

**{**

**j=partion(arr,low,high);**

**quick\_sort(arr,low,j-1);**

**quick\_sort(arr, j+1, high);**

**}**

**}**

**int main()**

**{**

**printf("Enter the Array size:");**

**scanf("%d",&n);**

**int i;**

**int arr[n];**

**printf("Enter the %d Elements in array:\n",n);**

**for (i = 0;i<n; i++)**

**scanf("%d",&arr[i]);**

**printf("Unsorted Array is:\n");**

**printf("[");**

**for (i = 0; i<n; i++)**

**printf(" %d ",arr[i]);**

**printf("]\n");**

**random\_shuffle(arr); //To randomize the array**

**quick\_sort(arr, 0,n-1); //function to sort the elements of array**

**printf("Randomized Quick sort array is:\n");**

**printf("[");**

**for (i = 0; i <n; i++)**

**printf(" %d ", arr[i]);**

**printf("]\n");**

**return 0;**

**}**

**Write program to implement to Monte Carlo Algorithm**

**Finding probability of 3 heads,6 heads,9 heads**

**SOURCE CODE:**

**//probability of 3 heads,6 heads,9 heads**

**#include <stdio.h>**

**#include <stdlib.h>**

**int main()**

**{**

**printf("Monte carlo Algorithm\n");**

**printf("Finding probability of 3 heads 6 heads and 9 heads\n");**

**int j,count=0,limit,i=0,head,tail;**

**float y,answer,h=2.5;**

**printf("Enter the limit\n");**

**scanf("%d",&limit);**

**while(i<=limit)**

**{**

**i++;**

**head=0,tail=0;**

**for(j=0;j<10;j++)**

**{**

**y=(float)((rand()% 65535)/65535.0f)\*10;**

**printf("%f\n",y);**

**if(y<h)**

**head=head+1;**

**else**

**tail=tail+1;**

**}**

**if((head==3)||(head==6)||(head==9))**

**count=count+1;**

**}**

**answer=(float)count/limit;**

**printf("The limit is %d \nThe probability is %f\n",limit,answer);**

**return 0;**

**}**

**Primality Testing**

**#include <stdio.h>**

**#include <stdlib.h>**

**#define ll long long**

**ll modulo(ll base, ll exponent, ll mod)**

**{**

**ll x = 1;**

**ll y = base;**

**while (exponent > 0)**

**{**

**if (exponent % 2 == 1)**

**x = (x \* y) % mod;**

**y = (y \* y) % mod;**

**exponent = exponent / 2;**

**}**

**return x % mod;**

**}**

**/\***

**\* Fermat's test for checking primality**

**\*/**

**int Fermat(ll p, int iterations)**

**{**

**int i;**

**if (p == 1)**

**{**

**return 0;**

**}**

**for (i = 0; i < iterations; i++)**

**{**

**ll a = rand() % (p - 1) + 1;**

**if (modulo(a, p - 1, p) != 1)**

**{**

**return 0;**

**}**

**}**

**return 1;**

**}**

**int main()**

**{**

**int iteration = 50;**

**ll num;**

**printf("Enter integer to test primality: ");**

**scanf("%lld", &num);**

**if (Fermat(num, iteration) == 1)**

**printf("%lld is prime ", num);**

**else**

**printf("%lld is not prime ", num);**

**return 0;**

**}**