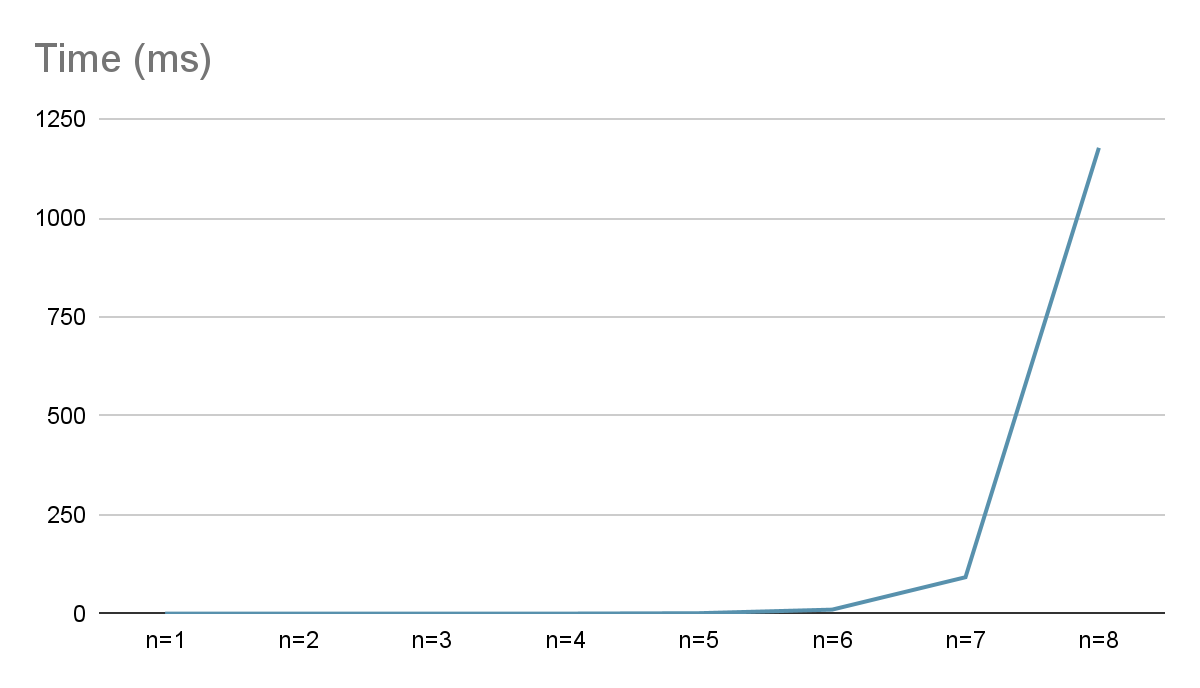
**OBSERVATION TABLE**

| **Value of N** | **Time in Milliseconds** |
| --- | --- |
| 1 | 0 |
| 2 | 0 |
| 3 | 0 |
| 4 | 0 |
| 5 | 1 |
| 6 | 10 |
| 7 | 92 |
| 8 | 1178 |

For n=9, we get runtime error(segmentation fault), estimating a very high time complexity for n=9.

From the observation table, we estimate the time complexity of the algorithm to be **O((n^2)\*n!)**

****

**Permutation Code-**

| #include <bits/stdc++.h> using namespace std; using namespace std::chrono; int n; void **rev\_ind**(int a[]) {  int rev[n];  int ind=0,reversal=0;  for(int i=0;i<n;i++)  rev[i]=0;  for(int i=0;i<n;i++)  {   for(int j=i+1;j<n;j++)  {  if(a[j]<a[i])  {  rev[a[j]-1]++;  reversal++;  }  }  if(i+1<n&&a[i]>a[i+1])  ind+=i+1;  }  cout<<"No of reversals = "<<reversal<<"\n";  cout<<"Reversal array: ";  for(int i=0;i<n;i++)  cout<<rev[i]<<" ";  cout<<"\n";  cout<<"Indicator = "<<ind<<"\n\n";  } void **getPerm**(int a[], int d[]) {   int mini=INT\_MIN;  int pos=-1;   for(int i=0;i<n;i++)  {   if(d[i]!=0&&a[i]>mini)  {  pos=i;  mini=a[i];  }  cout<<a[i]<<" ";  }    cout<<"\n";  rev\_ind(a);   if(mini==INT\_MIN)  return;   int mpos;  if(d[pos]==-1)  {   swap(a[pos],a[pos-1]);  swap(d[pos],d[pos-1]);  mpos=pos-1;  }  else  {  swap(a[pos],a[pos+1]);  swap(d[pos],d[pos+1]);  mpos=pos+1;  }   if(mpos==n-1||mpos==0)  d[mpos]=0;  else if(d[mpos]==-1&&a[mpos-1]>a[mpos])  d[mpos]=0;  else if(d[mpos]==1&&a[mpos+1]>a[mpos])  d[mpos]=0;   for(int i=0;i<n;i++)  {  if(a[i]>mini)  {  if(i<mpos)  d[i]=1;  else if(i>mpos)  d[i]=-1;  }   }  getPerm(a,d);  } int **main**() {    cout<<"enter a number n to generate permutation: ";  cin>>n;  auto start = high\_resolution\_clock::now();  int a[n];  int d[n];   for(int i=0;i<n;i++)  a[i]=i+1;   d[0]=0;  for(int i=1;i<n;i++)  d[i]=-1;   getPerm(a,d);  auto stop = high\_resolution\_clock::now();  auto duration = duration\_cast<microseconds>(stop - start);   cout << "Time taken for n="<<n<<": "<< duration.count()/1000 <<" milliseconds"<<"\n";   } |
| --- |

