Week 4

**Recursion/Recursion removal using Stack**

**Avni Arora\_20103153\_B6\_week#4**

1)

#include<bits/stdc++.h>

using namespace std;

stack<int> sorting(stack<int>s,int c)

{

if(s.empty()||s.top()<=c)

{

s.push(c);

return s;

}

else

{

int temp=s.top();

s.pop();

s=sorting(s,c);

s.push(temp);

}

return s;

}

stack<int> removes(stack<int>s)

{

if(!s.empty())

{

int c=s.top();

s.pop();

s=removes(s);

s=sorting(s,c);

}

return s;

}

void display (stack<int> s)

{

while(!s.empty())

{

cout<<s.top()<<" ";

s.pop();

}

cout<<endl;

}

int main()

{

stack<int>s;

cout<<"enter elements in stack : (press 0 to stop): ";

while(1)

{

int a;

cin>>a;

if(a!=0)

s.push(a);

else

break;

}

cout<<"given stack without sorting: ";

display(s);

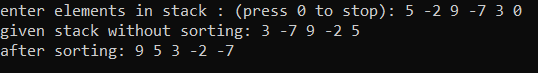
cout<<"after sorting: ";

s=removes(s);

display(s);

}

Output:



2)

#include<bits/stdc++.h>

using namespace std;

int isPalindrome(string s, int l, int h)

{

int flag=1;

while (l < h)

{

if (s[l] != s[h])

{

flag=0;

break;

}

l++;

h--;

}

return flag;

}

void palindrome(vector<string> &v,int index, int n, string s)

{

if (index >= n)

{

for(int i=0;i<v.size();i++)

{

cout<<v[i]<<" ";

}

cout<<endl;

return;

}

for (int i=index; i<n; i++)

{

if (isPalindrome(s,index, i))

{

v.push\_back(s.substr(index, i+1-index));

palindrome(v, i+1, n, s);

v.pop\_back();

}

}

}

int main()

{

cout<<"enter the string: ";

string s;

cin>>s;

int n = s.length();

vector<string>v;

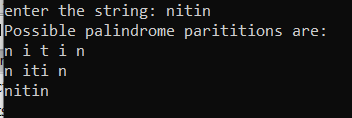
cout<<"Possible palindrome parititions are: "<<endl;

palindrome(v, 0, n, s);

return 0;

}

Output:



3)

#include<bits/stdc++.h>

using namespace std;

char \* reverse (stack<char> st)

{

char s[st.size()];

int i=0;

while(!st.empty())

{

s[i++]=st.top();

st.pop();

}

s[i]='\0';

cout<<s;

}

int main()

{

stack<char> st;

string s;

cout<<"enter the string : ";

cin>>s;

for(int i=0;i<s.length();i++)

{

st.push(s[i]);

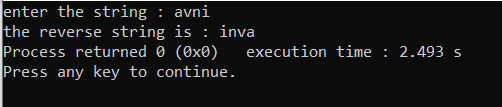
}

cout<<"the reverse string is : ";

reverse(st);

}

Output:



4)

#include<bits/stdc++.h>

using namespace std;

int disk(stack<int>& a,stack<int>& b){

if(b.empty()==true){

b.push(a.top());

a.pop();

return 1;

}

else if(a.empty()==true){

a.push(b.top());

b.pop();

return 0;

}

else{

if(b.top()>a.top()){

b.push(a.top());

a.pop();

return 1;

}

else{

a.push(b.top());

b.pop();

return 0;

}

}

}

void toh(int n)

{

stack<int> s,a,d;

int x=pow(2,n)-1;

int i=1;

for(int i=n;i>=1;i--)

{

s.push(i);

}

if(n%2==1)

{

while(i<=x)

{

if(i%3==1){

int j=disk(s,d);

if(j==1)

{

cout<<"Move the disk "<<d.top()<<" from source to destination"<<endl;

}

else

cout<<"Move the disk "<<s.top()<<" from destination to source"<<endl;

}

else if(i%3==2){

int j=disk(s,a);

if(j==1)

{

cout<<"Move the disk "<<a.top()<<" from source to auxiliary"<<endl;

}

else

cout<<"Move the disk "<<s.top()<<" from auxiliary to source"<<endl;

}

else{

int j=disk(a,d);

if(j==1)

{

cout<<"Move the disk "<<d.top()<<" from auxiliary to destination"<<endl;

}

else

cout<<"Move the disk "<<a.top()<<" from destination to auxiliary"<<endl;

}

i++;

}

}

else

{

while(i<=x)

{

if(i%3==1)

{

int j=disk(s,a);

if(j==1)

{

cout<<"Move the disk "<<a.top()<<" from source to auxiliary"<<endl;

}

else

cout<<"Move the disk "<<s.top()<<" from auxiliary to source"<<endl;

}

else if(i%3==2)

{

int j=disk(s,d);

if(j==1)

{

cout<<"Move the disk "<<d.top()<<" from source to destination"<<endl;

}

else

cout<<"Move the disk "<<s.top()<<" from destination to source"<<endl;

}

else

{

int j=disk(a,d);

if(j==1)

{

cout<<"Move the disk "<<d.top()<<" from auxiliary to destination"<<endl;

}

else

cout<<"Move the disk "<<a.top()<<" from destination to auxiliary"<<endl;

}

i++;

}

}

}

int main()

{

cout<<"enter no of discs: ";

int n;

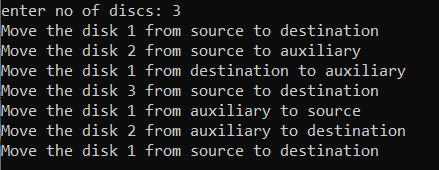
cin>>n;

toh(n);

return 0;

}

Output:



5)

#include<bits/stdc++.h>

using namespace std;

int Count=0;

int hailstone(int n)

{

if(n==1)

{

return Count+1;

}

else

{

Count++;

if(n%2==0)

{

cout<<n/2<<" ";

hailstone(n/2);

}

else

{

cout<<3\*n +1<<" ";

hailstone(3\*n+1);

}

}

}

int main()

{

int n;

cout<<"enter the number: ";

cin>>n;

cout<<"the hailstone sequence of given number: ";

cout<<n<<" ";

cout<<"\nthe length of the sequence is : "<<hailstone(n);

return 0;

}

Output:

