Chapter 5: Data Structures in Python

**1. Write a program to implement linked list**

l1=[]

l1.append('CP')

l1.append('JAVA')

l1.append('Python')

print('The existing list=',l1)

choice=0

while choice<5:

print('1-Add Element')

print('2-Remove Element')

print('3-Replace Element')

print('4-Search Element')

print('5-Exit')

choice=int(input('Enter Choice='))

if choice==1:

element=input('Enter element=')

pos=int(input('Enter position='))

l1.insert(pos,element)

print('List=',l1)

elif choice==2:

try:

element=input('Enter element to be removed=')

l1.remove(element)

except ValueError:

print('Element not found...')

else:

print('List=',l1)

elif choice==3:

element=input('Enter element=')

pos=int(input('Enter position='))

l1.pop(pos)

l1.insert(pos,element)

print('List=',l1)

elif choice==4:

element=input('Enter element to be searched=')

try:

pos=l1.index(element)

print('Element is present at position',pos)

except ValueError:

print('Element not present...')

else:

print('List=',l1)

else:

break

**2. Write a program to implement stack operations**

class stack:

def \_\_init\_\_(self):

self.st=[]

def isempty(self):

if self.st==[]:

return True

else:

return False

def push(self,element):

self.st.append(element)

def pop(self):

if self.isempty():

return -1

else:

return self.st.pop()

def peek(self):

n=len(self.st)

return self.st[n-1]

def search(self,element):

if self.isempty():

return -1

else:

try:

n=self.st.index(element)

return n

except ValueError:

return -2

def display(self):

return self.st

from stack import stack

s=stack()

choice=0

while choice<5:

print('1-Push Element')

print('2-Pop Element')

print('3-Peek Element')

print('4-Search Element')

print('5-Exit')

choice=int(input('Enter choice='))

if choice==1:

element=int(input('Enter element='))

s.push(element)

print('Stack=',s.display())

elif choice==2:

element=s.pop()

if element==-1:

print('Stack is empty')

else:

print('Popped element=',element)

print('Stack=',s.display())

elif choice==3:

element=s.peek()

print('Stack top element=',element)

elif choice==4:

element=int(input('Enter element to be searched='))

p=s.search(element)

if p==-1:

print('Stack is empty')

elif p==-2:

print('Element is not present')

else:

print('Element is present at position=',p)

else:

break

**3. Write a program to implement queue operations.**

class queue:

def \_\_init\_\_(self):

self.q=[]

def isempty(self):

if self.q==[]:

return True

else:

return False

def enqueue(self,element):

self.q.append(element)

def dequeue(self):

if self.isempty():

return -1

else:

return self.q.pop(0)

def search(self,element):

if self.isempty():

return -1

else:

try:

n=self.q.index(element)

return n

except ValueError:

return -2

def display(self):

return self.q

from queue import queue

q=queue()

choice=0

while choice<4:

print('1-Enqueue Element')

print('2-Dequeue Element')

print('3-Search Element')

print('4-Exit')

choice=int(input('Enter choice='))

if choice==1:

element=int(input('Enter element='))

q.enqueue(element)

print('Queue=',q.display())

elif choice==2:

element=q.pop()

if element==-1:

print('Queue is empty')

else:

print('Deleted element=',element)

print('Queue=',q.display())

elif choice==3:

element=int(input('Enter element to be searched='))

p=q.search(element)

if p==-1:

print('Queue is empty')

elif p==-2:

print('Element is not present')

else:

print('Element is present at position=',p)

else:

break