Національний технічний університет України «КПІ ім. Ігоря Сікорського»

Факультет Інформатики та Обчислювальної Техніки

Кафедра Автоматизованих Систем Обробки Інформації та Управління

Лабораторна робота № 8

з дисципліни «[Теорія алгоритмів](http://wiki.kpi.ua/index.php/%D0%A2%D0%B5%D0%BE%D1%80%D1%96%D1%8F%20%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D1%96%D0%B2_(20202020))»

на тему

«Бiнарнi дерева пошуку»

Виконав:

студент гр. ІС-91

Хмелiнiн Андрiй

Викладач:

[ст.вик. Халус О. А.](http://rozklad.kpi.ua/Schedules/ViewSchedule.aspx?v=318f7459-ddf0-47ee-b882-5650e756cf2a)

Київ – 2020

Код програми:

def InorderTreeWalk(ar, x=0): # записывает дерево в массив

global key, left, right

if x!=None:

InorderTreeWalk(ar, left[x])

ar.append(key[x])

InorderTreeWalk(ar, right[x])

def InorderTreeWrite(ar, x=0): # записывает бинарное дерево поиска в прошлое

global ii

if x!=None:

InorderTreeWrite(ar,left[x])

key[x]=ar[ii]

ii+=1

InorderTreeWrite(ar,right[x])

def TreeSort(): # вызывает функцию перевода бинарного дерева в массив, вызывает сортировку для этого массива, и вызывает функцию для записи отсортированого массива в дерево

array=[]

InorderTreeWalk(array)

array.sort()

InorderTreeWrite(array)

def SumFind(S,route,id,sum=0): # поиск элементов key которые вместе дают сумму S

global key,right,left, sums

if id != None:

if key[id]!=None:

sum+=key[id]

route.append(key[id]) # запись пути элементов в массив

if sum==S:

sums.append(route[:])

return

if sum >S:

return

SumFind(S, route[:], left[id],sum) # идем в левую ветку

SumFind(S,route[:],right[id],sum) # идем в правую ветку

inputname=input("Input file name (10c): ")

with open('Inputs/'+'input\_'+inputname+'.txt') as f: #считываем файл

InputData = f.read().split()

left=[None]\*len(InputData)

right=[None]\*len(InputData)

parent=[None]\*len(InputData)

key=[None]\*len(InputData)

sums=[]

S=int(input("Enter sum S: "))

ii=0

for i in range(len(InputData)):

InputData[i]=int(InputData[i])

def GoGrandParents(i,j): # находит индекс родителя для правого сына

global parent

if j!=0:

if i==right[parent[i]]:

i=parent[i]

else:

i=parent[i]

j-=1

i=GoGrandParents(i,j)

return i

for i in range(len(InputData)): # выставление связей между родителями и ветками

if InputData[i]!=0:

key[i]=InputData[i]

if i<len(InputData)-1:

if InputData[i+1]!=0:

left[i]=i+1

parent[i+1]=i

else:

if i<len(InputData)-1:

if InputData[i+1]!=0:

j=0

while InputData[i-1-j]==0:

j+=1

k=GoGrandParents(i-1-j,j)

right[k]=i+1

parent[i+1]=k

TreeSort()

for i in range(len(key)):

SumFind(S,[],i)

print(sums)

with open("Outputs/khmelinin\_output\_"+str(inputname)+"\_"+str(S)+".txt", 'w') as file: # запись в файл

for i in range(len(sums)):

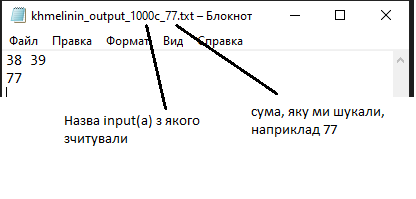
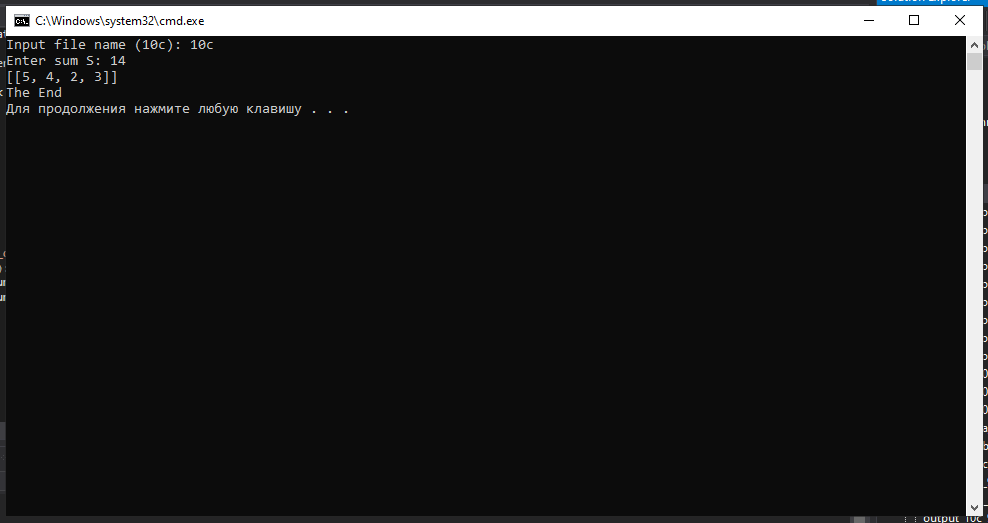
for j in range(len(sums[i])):

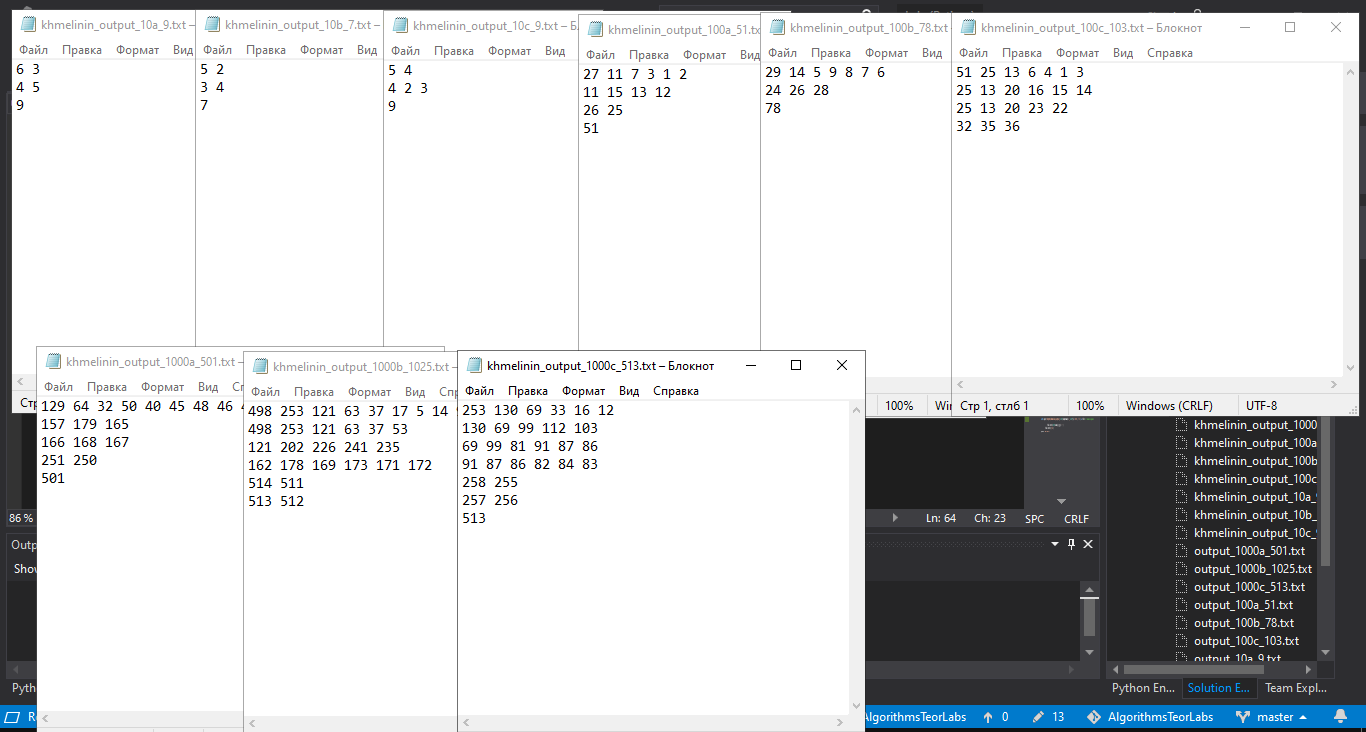
file.write(str(sums[i][j])+" ")

file.write("\n")

print("The End")

Приклад iнтерфейсу:





Висновок:

Я перетворив бінарне дерево у бінарне дерево пошуку. Також я реалiзував пошук сум послідовних вузлів в дереві. Сортування дерева у виглядi масива я реалiзовував стандартним вбудованим методом .sort() – гiбрид insert та merge sort.