Name: Prema Sunil Jadhar Sap 1d: 60004220127 Batch: C2-2 course: Advance Algorithm EXP-5 AIM: Implement KD-Tree THEORY: A KD-Tree also known as (k-dimensional true) is a binary search true where data in each node is a k-dimensional point in space. In short, it is a space partitioning data smother for organizing points in k-dimensional space. A non-leaf node in k-D tree divides the space in 2 parts, called as half-spaces. eg: creation of a 2-D True: (3,6), (17,15), (13,15), (6,12), (9,1), (2,7),(19,19) 3,6 2,71 17,15 13,15 6,12 10,19 9,1

Sundaran

	conclusion: K-D true have several advantages like efficient search. Dimensionality reduction, etc. It has the time and cpace complexity
	like efficient search. Dimensionality reduction,
	etc. It has the time and cpace complexity
	O DCn).
	Thus we studied and implemented K-DTue.
	A TANK
	and the supplemental states of the supplemental
	is a literate A Korina also though as litalined
	total mander early recept metanid to her least to the
22787	of this tensions with 4 is a box most to
	the skewe it is a space parhyboring the
	emility of soles entrine on mutuals
	mid sent and in what had have a wage
6.32	192-41 AN ELE BATTON HUNG & AT MARS AND
Cala N	(21,517, (21,717), (2,6) : 22,7 9-5 6 6 MONESSO
	1 3 (8)
	21 L1 S1 L1
	13,15,1-
	The state of the s
	FOR EDUCATIONAL USE
Sundaram	FOR EDUCATIONAL USE



Shri Vile Parle Kelavani Mandal's

DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING



(Autonomous College Affiliated to the University of Mumbai) NAAC Accredited with "A" Grade (CGPA: 3.18)

Academic Year: 2022-2023

Name:	Prerna Sunil Jadhav
Sap Id:	60004220127
Class:	T. Y. B. Tech (Computer Engineering)
Course:	Advance Algorithm Laboratory
Course Code:	DJ19CEL602
Experiment No.:	05

AIM: Implement KD-Tree.

CODE:

```
import math
class Node:
    def __init__(self, nums):
       self.nums = nums
        self.level = 0
        self.left = None
        self.right = None
def create_node(nums):
    return Node(nums)
def traverse_in_order(curr):
   if curr is None:
        return
    traverse_in_order(curr.left)
    print(f"({', '.join(map(str, curr.nums))}) ", end="")
    traverse_in_order(curr.right)
def make_kd_tree(seq, depth=0):
    if len(seq) == 0:
        return None
    k = len(seq[0])
    dim = depth % k
    seq.sort(key=lambda x: x[dim])
   mid = len(seq) // 2
   mid_elem = seq[mid]
    root = create_node(mid_elem)
```



Shri Vile Parle Kelavani Mandal's

DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING



(Autonomous College Affiliated to the University of Mumbai) NAAC Accredited with "A" Grade (CGPA: 3.18)

Academic Year: 2022-2023

```
left_sub_arr = seq[:mid]
    right_sub_arr = seq[mid+1:]

root.level = depth
    root.left = make_kd_tree(left_sub_arr, depth+1)
    root.right = make_kd_tree(right_sub_arr, depth+1)

return root

if __name__ == "__main__":
    seq = [[6,2], [7,1], [2,9], [3,6], [4,8], [8,4], [5,3], [1,5], [9,5]]
    root = make_kd_tree(seq)
    print("Inorder Traversal: ",end='')
    traverse_in_order(root)
```

OUTPUT:

PS C:\Users\Jadhav\Documents\BTech\Docs\6th Sem\AA\Code> & C:\msys64\mingw64\bin\python.exe "c:\Users\Jadhav\Documents\BTech\Docs\6th Sem\AA\Code\KD_Tree.py"
Inorder Traversal: (1, 5) (3, 6) (4, 8) (2, 9) (5, 3) (6, 2) (7, 1) (8, 4) (9, 5)