Kartheeka.Repalle

192372289

CSE-AI

6/8/2024

Binary heap:

```
#include <stdio.h>
#include <stdlib.h>
// Define the structure for a binary heap node
typedef struct Node {
    int data;
    struct Node* left;
    struct Node* right;
} Node;

// Function to create a new node
Node* createNode(int data) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    newNode->data = data;
    newNode->left = NULL;
    newNode->right = NULL;
    return newNode;
}

// Function to insert a node into the binary heap
void insert(Node** root, int data) {
    Node* newNode = createNode(data);
    if (*root == NULL) {
```

```
        *root = newNode;
    } else {
        // Perform heapify up to maintain the heap property
        Node* current = *root;
        while (current != NULL) {
            if (data < current->data) {
                if (current->left == NULL) {
                    current->left = newNode;
                    break;
                }
                current = current->left;
            } else {
                if (current->right == NULL) {
                    current->right = newNode;
                    break;
                }
                current = current->right;
            }
        }
    }
}

// Function to print the binary heap
void printHeap(Node* root) {
    if (root == NULL) {
        return;
    }
    printf("%d ", root->data);
    printHeap(root->left);
    printHeap(root->right);
```

```
}

int main() {
    Node* root = NULL;
    insert(&root, 10);
    insert(&root, 20);
    insert(&root, 30);
    insert(&root, 40);
    insert(&root, 50);
    printHeap(root);
    return 0;
}
```

Output:

10 20 30 40 50

HEAP SORT:

```
#include <stdio.h>
void heapify(int arr[], int n, int i) {
    int largest = i;
    int left = 2 * i + 1;
    int right = 2 * i + 2;

    if (left < n && arr[left] > arr[largest]) {
        largest = left;
    }

    if (right < n && arr[right] > arr[largest]) {
```

```
        largest = right;
    }

    if (largest != i) {
        int temp = arr[i];
        arr[i] = arr[largest];
        arr[largest] = temp;

        heapify(arr, n, largest);
    }
}

void heapSort(int arr[], int n) {
    for (int i = n / 2 - 1; i >= 0; i--) {
        heapify(arr, n, i);
    }

    for (int i = n - 1; i >= 0; i--) {
        int temp = arr[0];
        arr[0] = arr[i];
        arr[i] = temp;

        heapify(arr, i, 0);
    }
}

void printArray(int arr[], int n) {
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
```

```
    printf("\n");
}

int main() {
    int arr[] = {12, 11, 13, 5, 6, 7};
    int n = sizeof(arr) / sizeof(arr[0]);

    printf("Original array: ");
    printArray(arr, n);

    heapSort(arr, n);

    printf("Sorted array: ");
    printArray(arr, n);

    return 0;
}
```

Output:

Original array: 12 11 13 5 6 7

Sorted array: 5 6 7 11 12 13