```cpp
#include <bits/stdc++.h>
using namespace std;
char tree[10];

int root(char key) {
    if(tree[0] != '\0') {
        cout << "Tree already had root" << endl;
    } else {
        tree[0] = key;
    }
    return 0;
}

int set_left(char key, int parent) {
    if(tree[parent] == '\0') {
        cout << "\nCan't set child at"
                << (parent * 2) + 1
                << " , no parent found";
    } else {
        tree[(parent * 2) + 1] = key;
    }
    return 0;
}

int set_right(char key, int parent) {
    if(tree[parent] == '\0') {
        cout << "\nCan't set child at"
                << (parent * 2) + 2
                << " , no parent found";
    } else {
        tree[(parent * 2) + 2] = key;
    }
    return 0;
}

int print_tree() {
    cout << "\n";
    for(int i = 0; i < 10; i++) {
        if(tree[i] != '\0') {
            cout << tree[i] << " ";
        }
        else {
            cout << "- ";
        }
    }
    return 0;
}
```

```
int main() {
    root('A');
    set_left('B', 0);
    set_right('C', 0);
    set_left('D', 1);
    set_right('E', 1);
    set_left('F', 2);
    print_tree();
    return 0;
}
```

```
PS C:\Users\Hp\Documents\workspace\CS60\Semester4\Strukdat\10\output> & .\'1.exe'

  A B C D E F - - - -
```

1. Lakukan penelurusan Kode program di atas, dan jelaskan kegunaanya diperuntukkan untuk apa?
   Kode digunakan untuk membuat struktur data tree dengan menggunakan array, yang setiap node memiliki indeks yang ditentukan berdasarkan posisinya, dengan left_child dari node indeks **i** berada pada **i*2 + 1** sedangkan untuk right_child berada di **i*2 + 2**. Dan untuk indeks yang tidak terpakai akan dicetak sebagai "- ".

2. Jelaskan fungsi yang ada pada potongan kode (snippet) nomor l, apa yang dilakukan fungsi tersebut?
   Fungsi potongan kode tersebut digunakan untuk membuat root pada indeks[0] jika ternyata sudah ada root pada indeks[0] maka program akan menampilkan pesan "Tree already had root". Dan jika ternyata tidak, maka program akan membuat rootnya pada indeks[0].

3. Lengkapi potongan kode (snippet) nomor 3, lakukan penelusuran dan jelaskan apa kegunaan dari fungsi tersebut?

```
int set_right(char key, int parent) {
    if(tree[parent] == '\0') {
        cout << "\nCan't set child at"
                << (parent * 2) + 2
                << " , no parent found";
    } else {
        tree[(parent * 2) + 2] = key;
    }
    return 0;
}
```

Potongan kode ini digunakan untuk membuat child di sebelah kanan di node parent i pada i*2 + 2.

4. Implementasikan Keseluruhan kode, dan jalankan dengan kriteria berikut ini:
   root adalah T (indeks ke-0)
   indeks ke I adalah V, indeks ke-2 adalah S, indeks ke-3 adalah W, indeks ke-4 adalah P, indeks ke-5 adalah B, indeks ke-6 adalah Q
   tampilkan hasilnya!...

```
int main() {
    root('T');
    set_left('V', 0);
    set_right('S', 0);
    set_left('W', 1);
    set_right('P', 1);
    set_left('B', 2);
    set_right('Q', 2);
    print_tree();
    return 0;
}
```

```
PS C:\Users\Hp\Documents\workspace\CS60\Semester4\Strukdat\10\output> & .\'1.exe'

  T V S W P B Q - - -
```

---

---

```
#include <iostream>
using namespace std;

void fun1(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){
        swap(arr[i], arr[largest]);
        fun1(arr, n, largest);
    }
}

void fun2(int arr[], int n){
    int startIdx = n / 2 - 1;
    for(int i = startIdx; i >= 0; i--){
        fun1(arr, n, i);
    }
}

void fun3(int arr[], int& n){
    int lastElement = arr[n - 1];
    arr[0] = lastElement;
    n--;
    fun1(arr, n, 0);
}

void printArray(int arr[], int n){
    for(int i = 0; i < n; i++){
        cout << arr[i] << " ";
    }
    cout << endl;
}

int main(){
    int arr[] ={2, 4, 5, 1, 6, 10, 13, 17, 15, 8, 9};
    int n = sizeof(arr) / sizeof(arr[0]);
    fun2(arr, n);
    printArray(arr, n);
    int arr2[] = {10, 5, 3, 6, 2, 4, 17};
    int n2 = sizeof(arr2) / sizeof(arr2[0]);
    fun3(arr2, n2);
    printArray(arr2, n2);
    return 0;
}
```

```
PS C:\Users\Hp\Documents\workspace\CS60\Semester4\Strukdat\10\output> & .\'B.exe'
17 15 13 4 9 10 5 1 2 8 6
17 5 3 6 2 4
```

1. Pahami potongan kode tersebut, kemudian jelaskan fungsi ini digunakan untuk apa?
   Kode tersebut digunakan untuk mengimplementasikan operasi heap (max-heap), dengan fungsi fun1 untuk menjaga struktur heap, fun2 untuk membangun heap dari array yang disediakan, dan fun3 untuk mengganti element terakhir menjadi root dan lalu mengurangi ukuran heapnya.

2. Pada snippet 1, jelaskan baris kode tersebut untuk apa?
   Kode ini digunakan untuk memastikan suatu node dan subtree nya memenuhi max-heap (parent ≥ child) dengan memperbaiki struktur heap dari node i ke bawah

3. Lengkapi snippet pada nomor 3, dan jelaskan maksud dan jelaskan maksud dari potongan kode tersebut

```
void fun3(int arr[], int& n){
    int lastElement = arr[n - 1];
    arr[0] = lastElement;
    n = n - 1;

    fun1(arr, n, 0);
}
```

   Fungsi ini digunakan untuk mengganti element terakhir menjadi root dan lalu mengurangi ukuran heapnya.

4. Implementasikan Keseluruhan kode, dan jalankan dengan kriteria berikut ini:

   int arr[ ] = {20, 16, 11, 2, 4, 5, 1, 6, 10, 13};
   int n = sizeof (arr) / sizeof (arr [0]) ;
   fun2 (arr, n) ;
   fun3 (arr, n) ;

   tampilkan hasilnya!...

```
#include <iostream>
using namespace std;

void fun1(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){
        swap(arr[i], arr[largest]);
```

```
        fun1(arr, n, largest);
    }
}

void fun2(int arr[], int n){
    int startIdx = n / 2 - 1;
    for(int i = startIdx; i >= 0; i--){
        fun1(arr, n, i);
    }
}

void fun3(int arr[], int& n){
    int lastElement = arr[n - 1];
    arr[0] = lastElement;
    n = n - 1;
    fun1(arr, n, 0);
}

void printArray(int arr[], int n){
    for(int i = 0; i < n; i++){
        cout << arr[i] << " ";
    }
    cout << endl;
}

int main() {
    int arr[] = {20, 16, 11, 2, 4, 5, 1, 6, 10, 13};
    int n = sizeof(arr) / sizeof(arr[0]);

    fun2(arr, n);
    cout << "Setelah fun2: ";
    printArray(arr, n);

    fun3(arr, n);
    cout << "Setelah fun3: ";
    printArray(arr, n);

    return 0;
}
```

```
Setelah fun2: 20 16 11 10 13 5 1 6 2 4
Setelah fun3: 16 13 11 10 4 5 1 6 2
```