Laporan Ujian

Algoritma dan Pemrograman II

Disusun Oleh:

Nama: Dinar Fadilah

NIM: 231011401202

FAKULTAS ILMU KOMPUTER

PROGRAM STUDI TEKNIK INFORMATIKA

Tanggal: 08/01/2025	Algoritma dan Pemrogramman II

Source Code:

```
#include <iostream>
#include <queue>
#include <unordered_map>
#include <string>
#include <queue>
#include <unordered_map>
#include <string>

// Struktur node pohon Huffman
struct Node {
    char karakter;
    int frekuensi;
    Node* kiri;
    Node* kanan;
};

// Fungsi untuk membangun pohon Huffman
Node* buildHuffmanTree(const std::string& input) {
    // Buat tabel frekuensi karakter
    std::unordered_map<char, int> frekuensi;
    for (char c : input) {
        frekuensi[c]++;
    }

    // Buat antrian prioritas untuk node-node pohon Huffman
    std::priority_queue<Node*, std::vector<Node*>, std::function<bool(Node*, Node*)>>
antrian(
        [](Node* a, Node* b) { return a->frekuensi > b->frekuensi; });

    // Tambahkan node-node ke antrian
    for (const auto& pair : frekuensi) {
        Node* node = new Node{pair.first, pair.second, nullptr, nullptr};
        antrian.push(node);
    }

    // Bangun pohon Huffman
    while (antrian.size() > 1) {
        Node* node1 = antrian.top();
        antrian.pop();
        Node* node2 = antrian.top();
        antrian.pop();

        Node* newNode = new Node{'\0', node1->frekuensi + node2->frekuensi, node1,
node2};
        antrian.push(newNode);
    }

    return antrian.top();
}
```

```
// Fungsi untuk melakukan encoding Huffman
std::string encodeHuffman(const std::string& input, Node* root) {
    std::unordered_map<char, std::string> kode;
    std::function<void(Node*, std::string)> traverse = [&](Node* node, std::string
kodeSementara) {
        if (node->kiri == nullptr && node->kanan == nullptr) {
            kode[node->karakter] = kodeSementara;
        } else {
            traverse(node->kiri, kodeSementara + "0");
            traverse(node->kanan, kodeSementara + "1");
        }
    };

    traverse(root, "");

    std::string encoded;
    for (char c : input) {
        encoded += kode[c];
    }

    return encoded;
}

// Fungsi untuk melakukan decoding Huffman
std::string decodeHuffman(const std::string& encoded, Node* root) {
    std::string decoded;
    Node* currentNode = root;

    for (char c : encoded) {
        if (c == '0') {
            currentNode = currentNode->kiri;
        } else {
            currentNode = currentNode->kanan;
        }

        if (currentNode->kiri == nullptr && currentNode->kanan == nullptr) {
            decoded += currentNode->karakter;
            currentNode = root;
        }
    }

    return decoded;
}

int main() {
    cout << "Nama : Dinar Fadilah " << endl;
    cout << "NIM : 231011401202 " << endl;
    std::string input;
    std::cout << "Masukkan string: ";
    std::cin >> input;

    Node* root = buildHuffmanTree(input);
    std::string encoded = encodeHuffman(input, root);
```

```
    std::string decoded = decodeHuffman(encoded, root);

    std::cout << "String asli: " << input << std::endl;
    std::cout << "String terkompresi: " << encoded << std::endl;
    std::cout << "String dekompresi: " << decoded << std::endl;

    return 0;
}
```

PENJELASAN

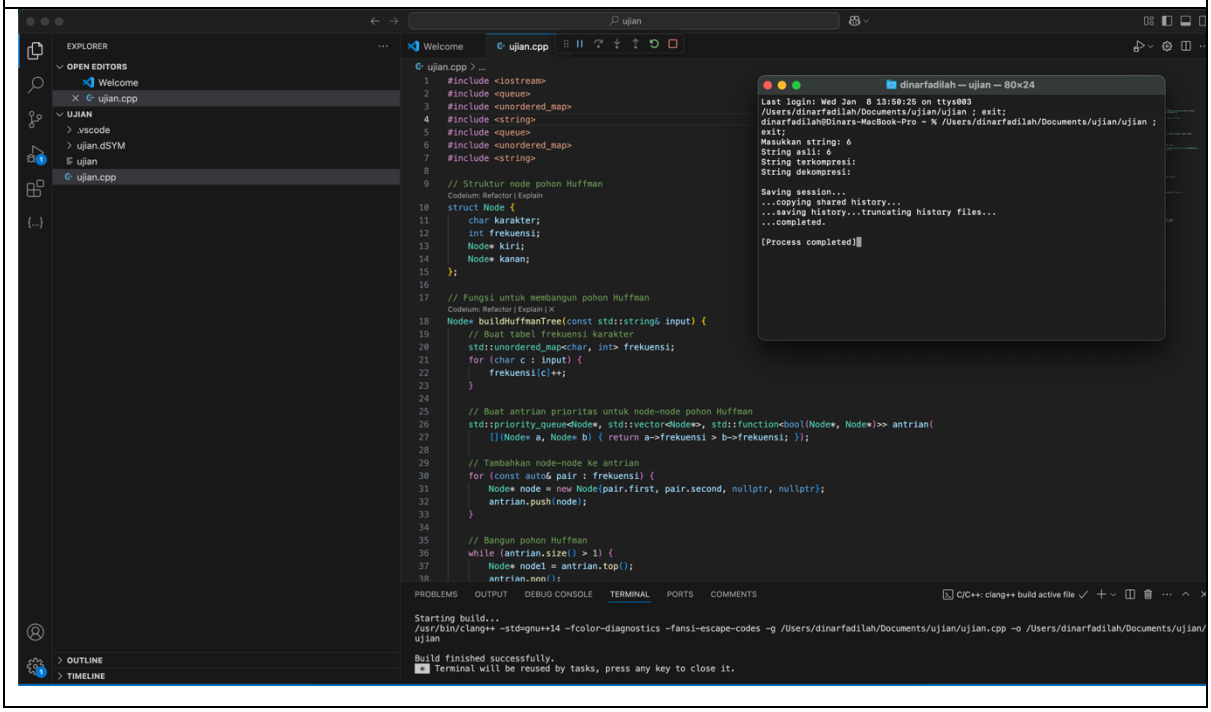
Pohon Huffman dibangun dengan cara berikut:

1. Buat tabel frekuensi karakter dari input string.
2. Buat antrian prioritas untuk node-node pohon Huffman, dengan prioritas berdasarkan frekuensi karakter.
3. Tambahkan node-node ke antrian, dengan setiap node mewakili karakter dan frekuensinya.
4. Bangun pohon Huffman dengan cara menggabungkan dua node dengan frekuensi terendah, sampai hanya ada satu node yang tersisa.

Proses encoding Huffman dilakukan dengan cara berikut:

1. Buat tabel kode Huffman, yang memetakan setiap karakter ke kode binernya.
2. Lakukan traversasi pohon Huffman, dengan setiap node yang dikunjungi menambahkan kode biner ke tabel kode.
3. Lakukan encoding string input dengan menggantikan setiap karakter dengan kode binernya.

HASIL


EXPLORER
OPEN EDITORS
Welcome
ujian.cpp
UJIAN
.vscode
ujian.dSYM
ujian
ujian.cpp
Welcome
ujian.cpp
ujian.cpp > ...
#include <iostream>
#include <queue>
#include <unordered_map>
#include <string>
#include <queue>
#include <unordered_map>
#include <string>
// Struktur node pohon Huffman
Codeium: Refactor | Explain
struct Node {
char karakter;
int frekuensi;
Node* kiri;
Node* kanan;
};
// Fungsi untuk membangun pohon Huffman
Codeium: Refactor | Explain | X
Node* buildHuffmanTree(const std::string& input) {
// Buat tabel frekuensi karakter
std::unordered_map<char, int> frekuensi;
for (char c : input) {
frekuensi[c]++;
}
// Buat antrian prioritas untuk node-node pohon Huffman
std::priority_queue<Node*, std::vector<Node*>, std::function<bool(Node*, Node*)>> antrian(
[](Node* a, Node* b) { return a->frekuensi > b->frekuensi; });
// Tambahkan node-node ke antrian
for (const auto& pair : frekuensi) {
Node* node = new Node{pair.first, pair.second, nullptr, nullptr};
antrian.push(node);
}
// Bangun pohon Huffman
while (antrian.size() > 1) {
Node* node1 = antrian.top();
dinarfadilah — ujian — 80×24
Last login: Wed Jan 8 13:50:25 on ttys003
/Users/dinarfadilah/Documents/ujian/ujian ; exit;
dinarfadilah@Dinars-MacBook-Pro ~ % /Users/dinarfadilah/Documents/ujian/ujian ;
exit;
Masukkan string: 6
String asli: 6
String terkompresi:
String dekompresi:
Saving session...
...copying shared history...
...saving history...truncating history files...
...completed.
[Process completed]
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
COMMENTS
C/C++: clang++ build active file
Starting build...
/usr/bin/clang++ -std=gnu++14 -fcolor-diagnostics -fansi-escape-codes -g /Users/dinarfadilah/Documents/ujian/ujian.cpp -o /Users/dinarfadilah/Documents/ujian/
ujian
Build finished successfully.
Terminal will be reused by tasks, press any key to close it.
OUTLINE
TIMELINE

Tanggal: 08/01/2025	Algoritma dan Pemrogramman II

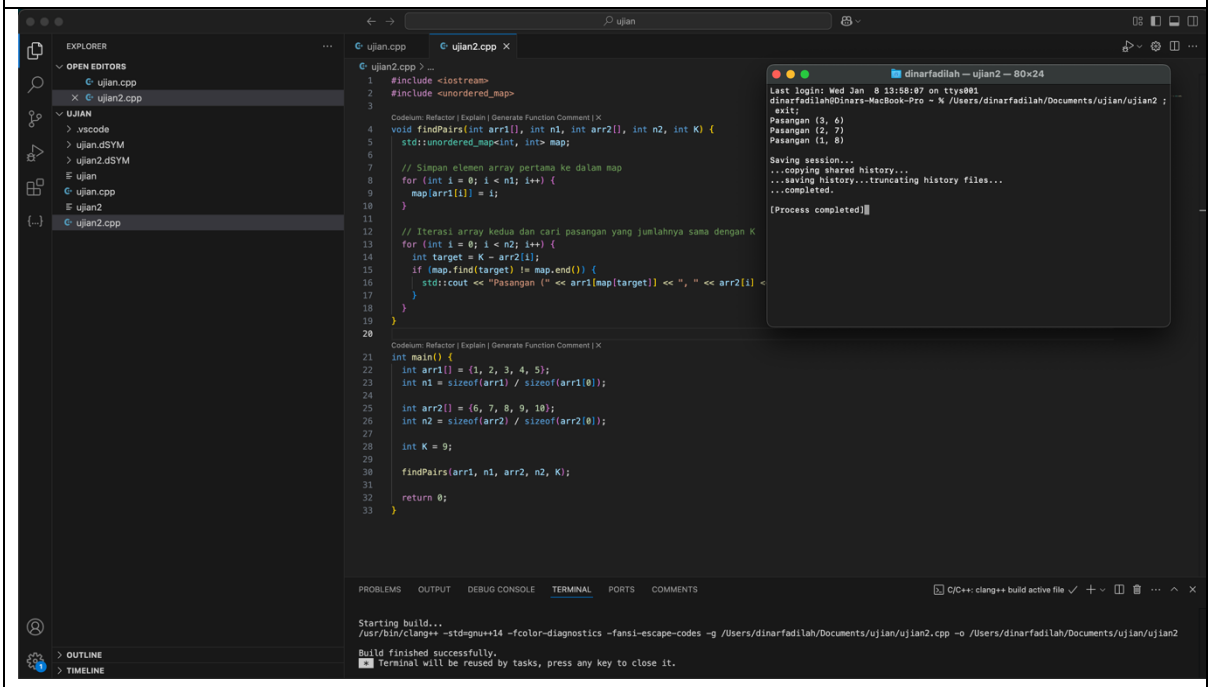
Source Code:

```
#include <iostream>
#include <unordered_map>

void findPairs(int arr1[], int n1, int arr2[], int n2, int K) {
  std::unordered_map<int, int> map;

  // Simpan elemen array pertama ke dalam map
  for (int i = 0; i < n1; i++) {
    map[arr1[i]] = i;
  }

  // Iterasi array kedua dan cari pasangan yang jumlahnya sama dengan K
  for (int i = 0; i < n2; i++) {
    int target = K - arr2[i];
    if (map.find(target) != map.end()) {
      std::cout << "Pasangan (" << arr1[map[target]] << ", " << arr2[i] << ")" << std::endl;
    }
  }
}

int main() {
  int arr1[] = {1, 2, 3, 4, 5};
  int n1 = sizeof(arr1) / sizeof(arr1[0]);

  int arr2[] = {6, 7, 8, 9, 10};
  int n2 = sizeof(arr2) / sizeof(arr2[0]);

  int K = 9;

  findPairs(arr1, n1, arr2, n2, K);

  return 0;
}
```

PENJELASAN

Algoritma ini menggunakan struktur data **unordered_map** untuk menyimpan elemen array pertama sebagai kunci dan indeksnya sebagai nilai. Kemudian, algoritma iterasi array kedua dan mencari pasangan yang jumlahnya sama dengan K dengan menggunakan fungsi **find** pada map.
Kompleksitas waktu algoritma ini adalah O(n1 + n2), karena kita iterasi array pertama sekali dan array kedua sekali. Kompleksitas ruang algoritma ini adalah O(n1), karena kita menggunakan map untuk menyimpan elemen array pertama.

HASIL

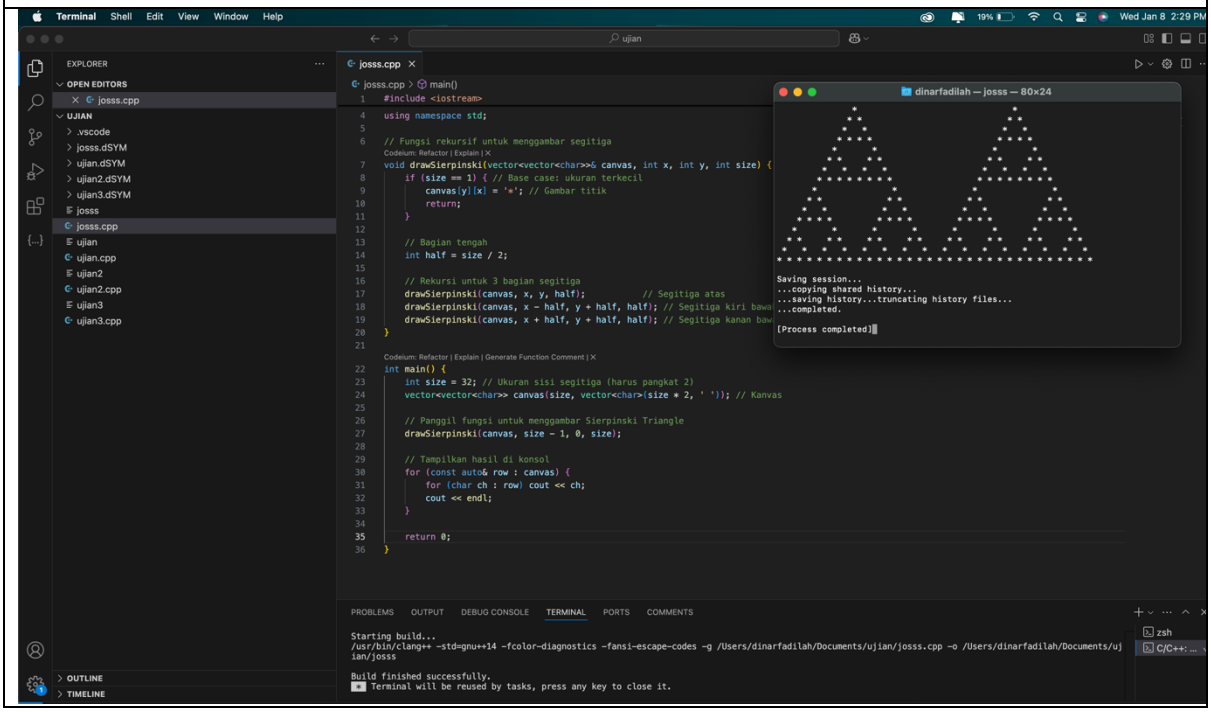

EXPLORER
OPEN EDITORS
ujian.cpp
ujian2.cpp
UJIAN
.vscode
ujian.dSYM
ujian2.dSYM
ujian
ujian.cpp
ujian2
ujian2.cpp
OUTLINE
TIMELINE
ujian.cpp
ujian2.cpp
ujian2.cpp > ...
#include <iostream>
#include <unordered_map>
Codeium: Refactor | Explain | Generate Function Comment | ×
void findPairs(int arr1[], int n1, int arr2[], int n2, int K) {
std::unordered_map<int, int> map;
// Simpan elemen array pertama ke dalam map
for (int i = 0; i < n1; i++) {
map[arr1[i]] = i;
}
// Iterasi array kedua dan cari pasangan yang jumlahnya sama dengan K
for (int i = 0; i < n2; i++) {
int target = K - arr2[i];
if (map.find(target) != map.end()) {
std::cout << "Pasangan (" << arr1[map[target]] << ", " << arr2[i] <
}
}
}
Codeium: Refactor | Explain | Generate Function Comment | ×
int main() {
int arr1[] = {1, 2, 3, 4, 5};
int n1 = sizeof(arr1) / sizeof(arr1[0]);
int arr2[] = {6, 7, 8, 9, 10};
int n2 = sizeof(arr2) / sizeof(arr2[0]);
int K = 9;
findPairs(arr1, n1, arr2, n2, K);
return 0;
}
dinarfadilah — ujian2 — 80×24
Last login: Wed Jan 8 13:58:07 on ttys001
dinarfadilah@Dinars-MacBook-Pro ~ % /Users/dinarfadilah/Documents/ujian/ujian2 ;
exit;
Pasangan (3, 6)
Pasangan (2, 7)
Pasangan (1, 8)
Saving session...
...copying shared history...
...saving history...truncating history files...
...completed.
[Process completed]
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
COMMENTS
C/C++: clang++ build active file
Starting build...
/usr/bin/clang++ -std=gnu++14 -fcolor-diagnostics -fansi-escape-codes -g /Users/dinarfadilah/Documents/ujian/ujian2.cpp -o /Users/dinarfadilah/Documents/ujian/ujian2
Build finished successfully.
Terminal will be reused by tasks, press any key to close it.

Tanggal: 08/01/2025	Algoritma dan Pemrogramman II

Source Code:

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

// Fungsi rekursif untuk menggambar segitiga
void drawSierpinski(vector<vector<char>>& canvas, int x, int y, int size) {
    if (size == 1) { // Base case: ukuran terkecil
        canvas[y][x] = '*'; // Gambar titik
        return;
    }

    // Bagian tengah
    int half = size / 2;

    // Rekursi untuk 3 bagian segitiga
    drawSierpinski(canvas, x, y, half);           // Segitiga atas
    drawSierpinski(canvas, x - half, y + half, half); // Segitiga kiri bawah
    drawSierpinski(canvas, x + half, y + half, half); // Segitiga kanan bawah
}

int main() {
    int size = 32; // Ukuran sisi segitiga (harus pangkat 2)
    vector<vector<char>> canvas(size, vector<char>(size * 2, ' ')); // Kanvas

    // Panggil fungsi untuk menggambar Sierpinski Triangle
    drawSierpinski(canvas, size - 1, 0, size);

    // Tampilkan hasil di konsol
    for (const auto& row : canvas) {
        for (char ch : row) cout << ch;
        cout << endl;
    }

    return 0;
}
```

HASIL

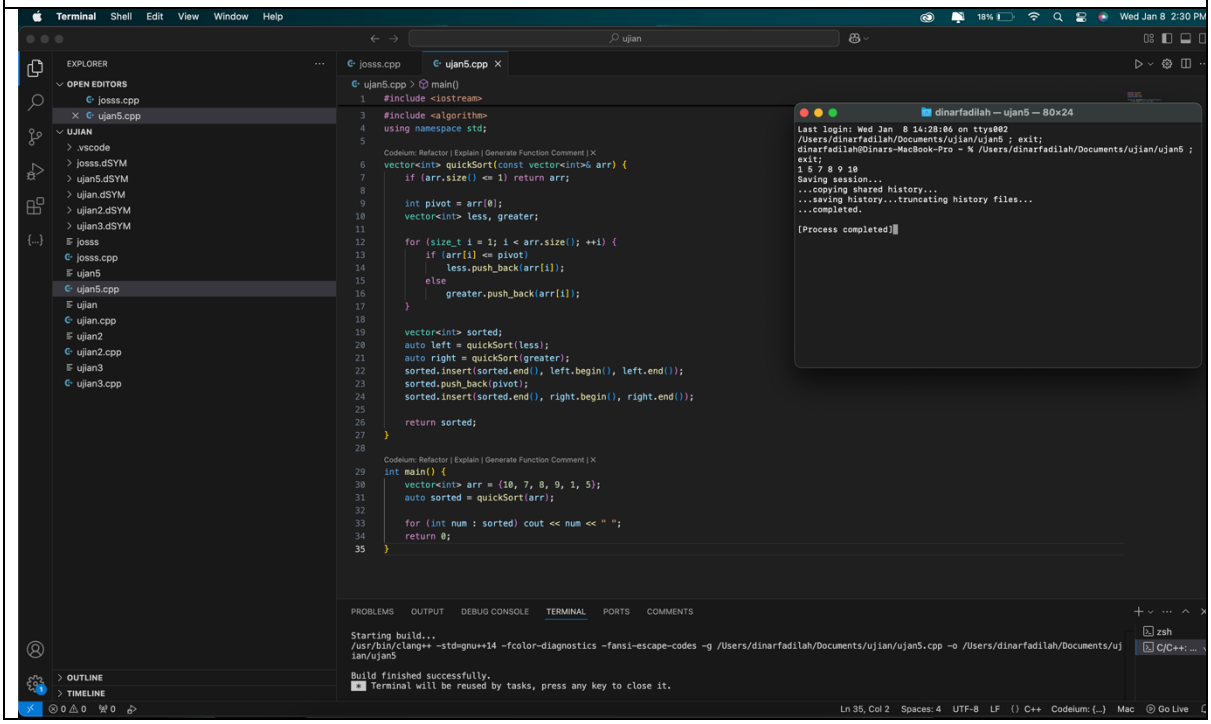

Terminal Shell Edit View Window Help
Wed Jan 8 2:29 PM
EXPLORER
OPEN EDITORS
josss.cpp
UJIAN
.vscode
josss.dSYM
ujian.dSYM
ujian2.dSYM
ujian3.dSYM
josss
josss.cpp
ujian
ujian.cpp
ujian2
ujian2.cpp
ujian3
ujian3.cpp
OUTLINE
TIMELINE
josss.cpp > main()
#include <iostream>
using namespace std;
// Fungsi rekursif untuk menggambar segitiga
Codeium: Refactor | Explain |
void drawSierpinski(vector<vector<char>>& canvas, int x, int y, int size) {
if (size == 1) { // Base case: ukuran terkecil
canvas[y][x] = '*'; // Gambar titik
return;
}
// Bagian tengah
int half = size / 2;
// Rekursi untuk 3 bagian segitiga
drawSierpinski(canvas, x, y, half); // Segitiga atas
drawSierpinski(canvas, x - half, y + half, half); // Segitiga kiri bawa
drawSierpinski(canvas, x + half, y + half, half); // Segitiga kanan baw
}
Codeium: Refactor | Explain | Generate Function Comment |
int main() {
int size = 32; // Ukuran sisi segitiga (harus pangkat 2)
vector<vector<char>> canvas(size, vector<char>(size * 2, ' ')); // Kanvas
// Panggil fungsi untuk menggambar Sierpinski Triangle
drawSierpinski(canvas, size - 1, 0, size);
// Tampilkan hasil di konsol
for (const auto& row : canvas) {
for (char ch : row) cout << ch;
cout << endl;
}
return 0;
}
dinarfadilah — josss — 80×24
Saving session...
...copying shared history...
...saving history...truncating history files...
...completed.
[Process completed]
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
Starting build...
/usr/bin/clang++ -std=gnu++14 -fcolor-diagnostics -fansi-escape-codes -g /Users/dinarfadilah/Documents/ujian/josss.cpp -o /Users/dinarfadilah/Documents/ujian/josss
Build finished successfully.
Terminal will be reused by tasks, press any key to close it.
zsh
C/C++: ...

Tanggal: 08/01/2025	Algoritma dan Pemrogramman II

Source Code:

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

vector<int> quickSort(const vector<int>& arr) {
    if (arr.size() <= 1) return arr;

    int pivot = arr[0];
    vector<int> less, greater;

    for (size_t i = 1; i < arr.size(); ++i) {
        if (arr[i] <= pivot)
            less.push_back(arr[i]);
        else
            greater.push_back(arr[i]);
    }

    vector<int> sorted;
    auto left = quickSort(less);
    auto right = quickSort(greater);
    sorted.insert(sorted.end(), left.begin(), left.end());
    sorted.push_back(pivot);
    sorted.insert(sorted.end(), right.begin(), right.end());

    return sorted;
}

int main() {
    vector<int> arr = {10, 7, 8, 9, 1, 5};
    auto sorted = quickSort(arr);

    for (int num : sorted) cout << num << " ";
    return 0;
}
```

HASIL


Terminal Shell Edit View Window Help
Wed Jan 8 2:30 PM
EXPLORER
OPEN EDITORS
josss.cpp
ujan5.cpp
UJIAN
.vscode
josss.dSYM
ujan5.dSYM
ujian.dSYM
ujian2.dSYM
ujian3.dSYM
josss
josss.cpp
ujan5
ujan5.cpp
ujian
ujian.cpp
ujian2
ujian2.cpp
ujian3
ujian3.cpp
OUTLINE
TIMELINE
ujan5.cpp > main()
#include <iostream>
#include <algorithm>
using namespace std;
Codeium: Refactor | Explain | Generate Function Comment
vector<int> quickSort(const vector<int>& arr) {
if (arr.size() <= 1) return arr;
int pivot = arr[0];
vector<int> less, greater;
for (size_t i = 1; i < arr.size(); ++i) {
if (arr[i] <= pivot)
less.push_back(arr[i]);
else
greater.push_back(arr[i]);
}
vector<int> sorted;
auto left = quickSort(less);
auto right = quickSort(greater);
sorted.insert(sorted.end(), left.begin(), left.end());
sorted.push_back(pivot);
sorted.insert(sorted.end(), right.begin(), right.end());
return sorted;
}
Codeium: Refactor | Explain | Generate Function Comment
int main() {
vector<int> arr = {10, 7, 8, 9, 1, 5};
auto sorted = quickSort(arr);
for (int num : sorted) cout << num << " ";
return 0;
}
dinarfadilah — ujan5 — 80×24
Last login: Wed Jan 8 14:28:06 on ttys002
/Users/dinarfadilah/Documents/ujian/ujan5 ; exit;
dinarfadilah@Dinars-MacBook-Pro ~ % /Users/dinarfadilah/Documents/ujian/ujan5 ;
exit;
1 5 7 8 9 10
Saving session...
...copying shared history...
...saving history...truncating history files...
...completed.
[Process completed]
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
Starting build...
/usr/bin/clang++ -std=gnu++14 -fcolor-diagnostics -fansi-escape-codes -g /Users/dinarfadilah/Documents/ujian/ujan5.cpp -o /Users/dinarfadilah/Documents/ujian/ujan5
Build finished successfully.
Terminal will be reused by tasks, press any key to close it.
zsh
C/C++: ...
Ln 35, Col 2 Spaces: 4 UTF-8 LF C++ Codeium: {...} Mac Go Live

Tanggal: 08/01/2025	Algoritma dan Pemrogramman II

Source Code: