LAPORAN PRAKTIKUM

Modul 05

“SINGLE LINKED LIST (BAGIAN KEDUA)”

Disusun Oleh:

Muhammad Widya Tirta Cahyatama – 21104047

S1SE07-01

Dosen:

Yudha Islami Sulistya, S.Kom., M.Cs

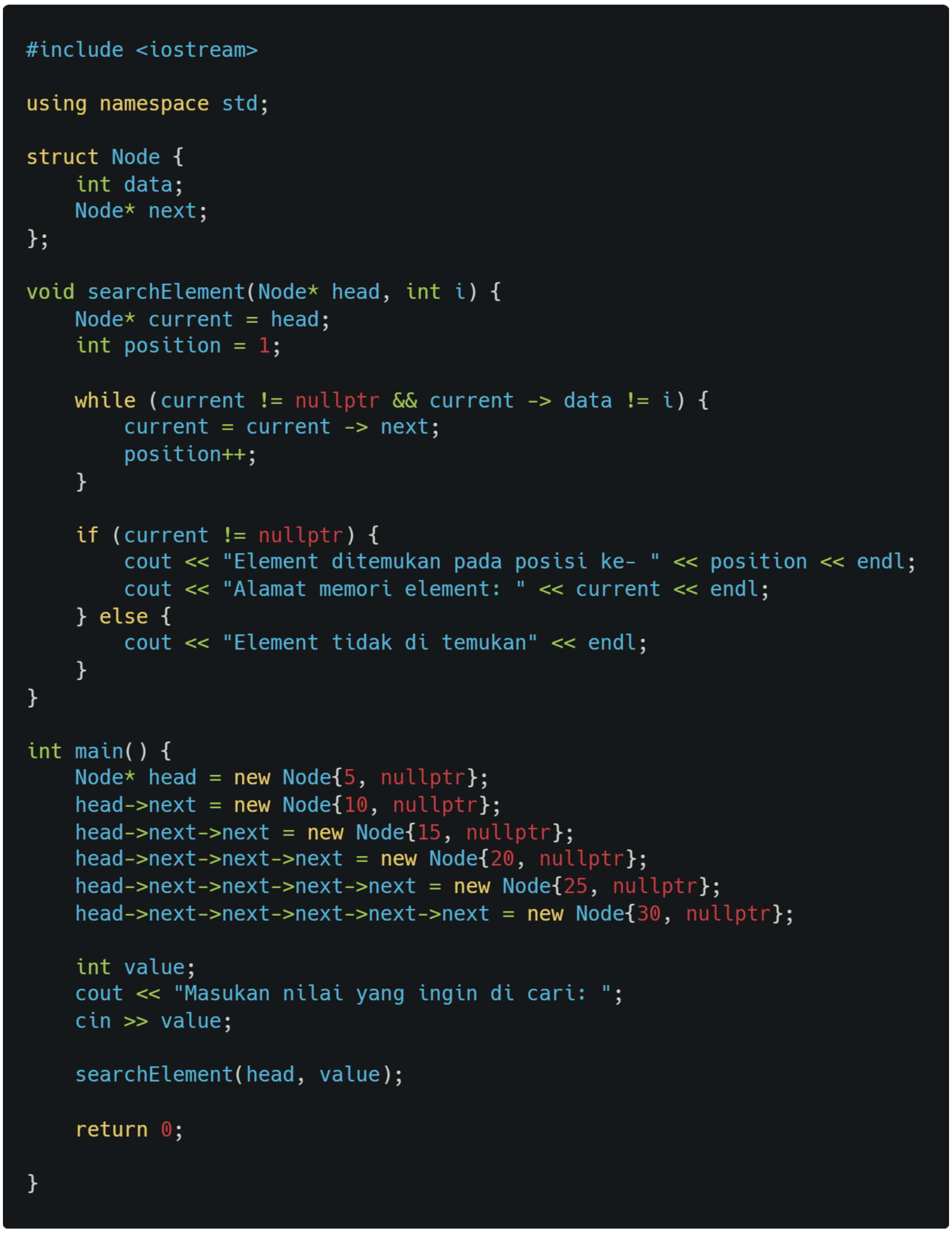
PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

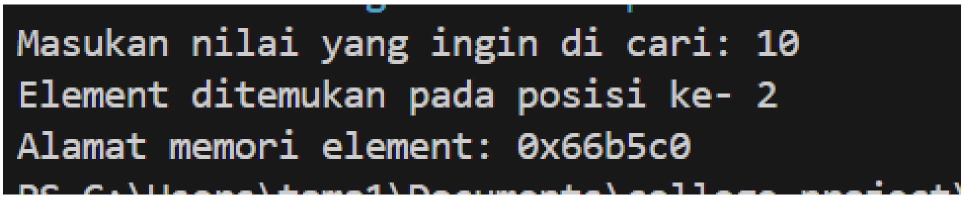
2024

Soal 1



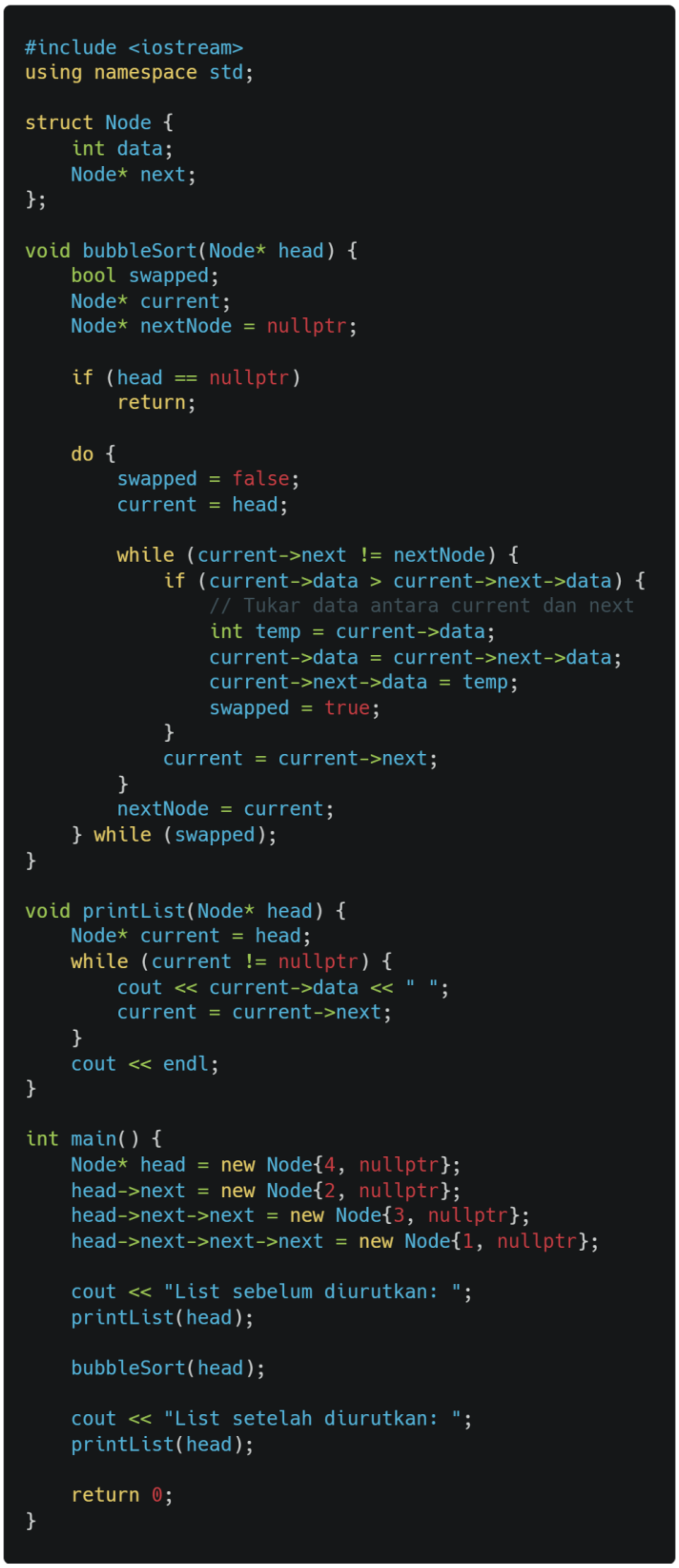
```cpp
#include <iostream>

using namespace std;

struct Node {
    int data;
    Node* next;
};

void searchElement(Node* head, int i) {
    Node* current = head;
    int position = 1;

    while (current != nullptr && current -> data != i) {
        current = current -> next;
        position++;
    }

    if (current != nullptr) {
        cout << "Element ditemukan pada posisi ke- " << position << endl;
        cout << "Alamat memori element: " << current << endl;
    } else {
        cout << "Element tidak di temukan" << endl;
    }
}

int main() {
    Node* head = new Node{5, nullptr};
    head->next = new Node{10, nullptr};
    head->next->next = new Node{15, nullptr};
    head->next->next->next = new Node{20, nullptr};
    head->next->next->next->next = new Node{25, nullptr};
    head->next->next->next->next->next = new Node{30, nullptr};

    int value;
    cout << "Masukan nilai yang ingin di cari: ";
    cin >> value;

    searchElement(head, value);

    return 0;

}
```

Output:



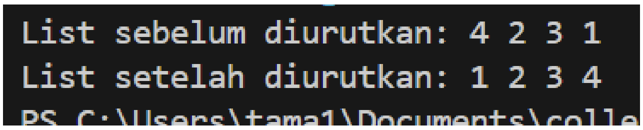
```
Masukan nilai yang ingin di cari: 10
Element ditemukan pada posisi ke- 2
Alamat memori element: 0x66b5c0
```

Soal 2



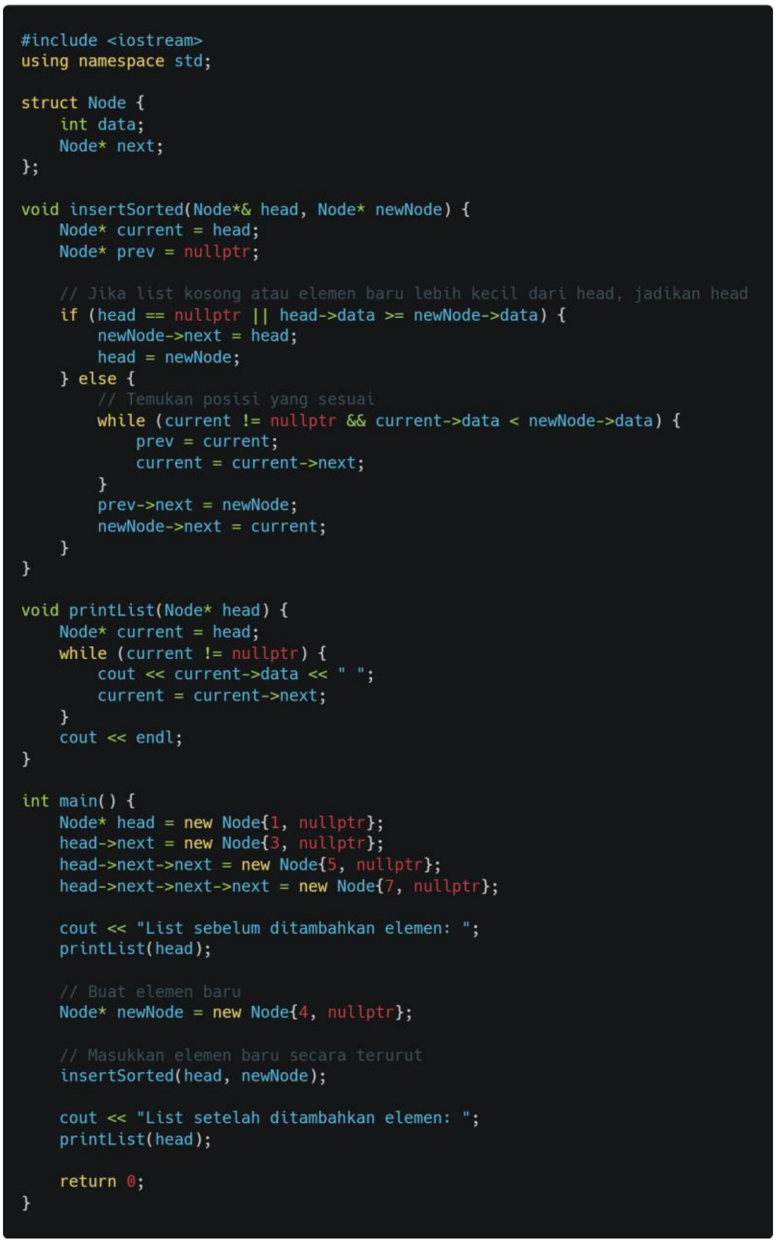
```cpp
#include <iostream>
using namespace std;

struct Node {
    int data;
    Node* next;
};

void bubbleSort(Node* head) {
    bool swapped;
    Node* current;
    Node* nextNode = nullptr;

    if (head == nullptr)
        return;

    do {
        swapped = false;
        current = head;

        while (current->next != nextNode) {
            if (current->data > current->next->data) {
                // Tukar data antara current dan next
                int temp = current->data;
                current->data = current->next->data;
                current->next->data = temp;
                swapped = true;
            }
            current = current->next;
        }
        nextNode = current;
    } while (swapped);
}

void printList(Node* head) {
    Node* current = head;
    while (current != nullptr) {
        cout << current->data << " ";
        current = current->next;
    }
    cout << endl;
}

int main() {
    Node* head = new Node{4, nullptr};
    head->next = new Node{2, nullptr};
    head->next->next = new Node{3, nullptr};
    head->next->next->next = new Node{1, nullptr};

    cout << "List sebelum diurutkan: ";
    printList(head);

    bubbleSort(head);

    cout << "List setelah diurutkan: ";
    printList(head);

    return 0;
}
```

Output:



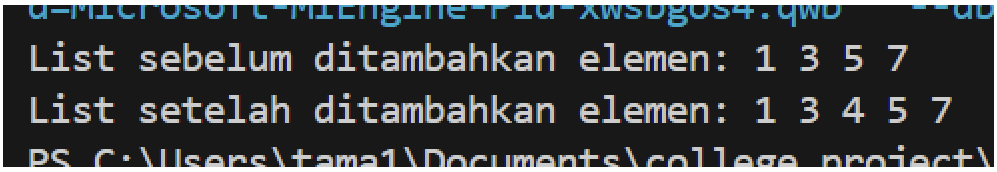
```
List sebelum diurutkan: 4 2 3 1
List setelah diurutkan: 1 2 3 4
PS C:\Users\tama1\Documents\colle
```

Soal 3



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

struct Node {
    int data;
    Node* next;
};

void insertSorted(Node*& head, Node* newNode) {
    Node* current = head;
    Node* prev = nullptr;

    // Jika list kosong atau elemen baru lebih kecil dari head, jadikan head
    if (head == nullptr || head->data >= newNode->data) {
        newNode->next = head;
        head = newNode;
    } else {
        // Temukan posisi yang sesuai
        while (current != nullptr && current->data < newNode->data) {
            prev = current;
            current = current->next;
        }
        prev->next = newNode;
        newNode->next = current;
    }
}

void printList(Node* head) {
    Node* current = head;
    while (current != nullptr) {
        cout << current->data << " ";
        current = current->next;
    }
    cout << endl;
}

int main() {
    Node* head = new Node{1, nullptr};
    head->next = new Node{3, nullptr};
    head->next->next = new Node{5, nullptr};
    head->next->next->next = new Node{7, nullptr};

    cout << "List sebelum ditambahkan elemen: ";
    printList(head);

    // Buat elemen baru
    Node* newNode = new Node{4, nullptr};

    // Masukkan elemen baru secara terurut
    insertSorted(head, newNode);

    cout << "List setelah ditambahkan elemen: ";
    printList(head);

    return 0;
}
```

Output:



```
d=Microsoft-MIEngine-Pid-xwsbgos4.qwb   --db
List sebelum ditambahkan elemen: 1 3 5 7
List setelah ditambahkan elemen: 1 3 4 5 7
PS C:\Users\tama1\Documents\college project\
```