Tugas Algoritma dan Struktur Data

PAS 5 & 6 – Single Linked List ( Delete, Insert )

**Nama : Ferdian Nur Fariza  
NIM : A11.2023.15074**

**Source Code**

|  |
| --- |
| #include <iostream>  using namespace std;  class Node  {  public:  int data;  Node \*next;  Node()  {  data = 0;  next = NULL;  }  Node(int data)  {  this->data = data;  this->next = NULL;  }  };  class Linkedlist  {  private:  Node \*head; // pointer yg ada di Node pertama  Node \*tail; // pointer yg ada di Node terakhir  public:  Linkedlist()  {  head = NULL;  tail = NULL;  }  void insertDepan(int value);  void insertBelakang(int value);  void cetak();  void hapusDepan();  void hapusBelakang();  int hitungNode();  bool searchData(int k);  bool searchData2(int k, int n);  };  void Linkedlist::insertDepan(int value)  {  Node \*temp = new Node(value); // memanggil konstruktor inputan  if (head == NULL)  { // jika list nya kosong  head = temp;  tail = temp;  }  else  {  temp->next = head;  head = temp;  }  }  void Linkedlist::insertBelakang(int value)  {  Node \*temp = new Node(value);  if (head == NULL)  {  head = temp;  tail = temp;  }  else  {  tail->next = temp;  tail = temp;  }  }  void Linkedlist::cetak()  {  Node \*temp = new Node;  temp = head;  while (temp != NULL)  {  if (temp->next != NULL)  {  cout << "[" << temp->data << "|" << temp->next << "]" << " --> ";  }  else  {  cout << "[" << temp->data << "|" << temp->next << "]";  }  temp = temp->next;  }  cout << endl;  }  void Linkedlist::hapusDepan()  {  Node \*temp = new Node;  temp = head;  head = head->next;  delete temp;  cetak(); // opsional  }  void Linkedlist::hapusBelakang()  {  Node \*current = new Node;  Node \*previous = new Node;  current = head;  while (current->next != NULL)  {  previous = current;  current = current->next;  }  tail = previous;  previous->next = NULL;  delete current;  cetak(); // opsional  }  int Linkedlist::hitungNode()  {  int s = 0;  Node \*temp = new Node;  temp = head;  while (temp != NULL)  {  s++;  temp = temp->next;  }  return s;  }  bool Linkedlist::searchData(int k)  {  // LinearSearch + Break  bool found = false;  Node \*temp = new Node;  temp = head;  while (temp != NULL)  {  if (temp->data == k)  {  found = true;  break;  }  temp = temp->next;  }  return found;  }  bool Linkedlist::searchData2(int k, int n)  {  // LinearSearch + Break  bool found = false;  Node \*temp = new Node;  temp = head;  for (int i = 1; i <= n; i++)  {  if (temp->data == k)  {  found = true;  break;  }  temp = temp->next;  }  return found;  }  int main()  {  cout << "Single Linked List Manual" << endl;  Node \*head = new Node;  head->data = 100;  head->next = new Node;  head->next->data = 80;  head->next->next = new Node(5);  // proses cetak  Node \*temp = new Node;  temp = head;  while (temp != NULL)  {  cout << temp->data << " ";  temp = temp->next;  }  cout << endl;  cout << "\nSingle Linked List memanggil class Linkedlist" << endl;  Linkedlist sll;  cout << "insertDepan (100)\n";  sll.insertDepan(100);  sll.cetak();  cout << "\ninsertDepan (1)\n";  sll.insertDepan(1);  sll.cetak();  cout << "\ninsertBelakang (84)\n";  sll.insertBelakang(84);  sll.cetak();  cout << "Nodenya ada berapa? " << sll.hitungNode() << endl;  cout << "Apakah ada data 100? " << sll.searchData(100) << endl;  cout << "Apakah ada data 33? " << sll.searchData(33) << endl;  int panjangNode = sll.hitungNode();  cout << "\nhapus Depan ()\n";  sll.hapusDepan();  cout << "\nhapusBelakang()\n";  sll.hapusBelakang();  cout << "Sekarang nodenya ada berapa? " << sll.hitungNode() << endl;  return 0;  } |

**Screenshot Output :**

