```
typedef struct Data
{
   int nilai;
   Data *next;
};
Data *head;
Data *tail;
void awal ()
{
   head=NULL;
}

bool isEmpty()
{
   if (head==NULL)
      return true;
   return false;
}

void tambahDataDepan (int DataBaru)
{
   Data *baru;
   baru = new Data;
   baru -> nilai = DataBaru;
   baru -> next = NULL;
   if (isEmpty())
   {
      head = baru;
      head->next=NULL;
   }
   else{baru->next = head; head = baru;}
   cout << "Data Depan" << DataBaru << " Masuk"
<< endl;
}
void tambahDataBelakang (int DataBaru)
{
   Data *baru, *bantu;
   baru = new Data;
   baru->nilai=DataBaru;
   baru->next = NULL;
   if (isEmpty())
   {  head=baru;
      head->next=NULL; }
   else
   {
      bantu=head;
      while(bantu->next!=NULL)
      {
         bantu=bantu->next;
      }
      bantu->next=baru;
   }
   cout << "Data Belakang " << DataBaru <<
"Masuk"  << endl;
}
```

```
void hapusDepan ()
{
   Data *hapus;
   int d;
   if (!isEmpty())
   {
      if(head->next !=NULL)
      {
         hapus = head;
         d = hapus->nilai;
         head = hapus->next;
         delete hapus;
      }
      else
      {
         d = head->nilai;
         head = NULL;
      }
      cout << d <<"Terhapus" << endl;
   }
   else cout << "Masih Kosong" << endl;
}


void hapusBelakang ()
{
   Data *hapus, *bantu;
   int tmp;
   if (!isEmpty())
   {
      if (head->next!=NULL)
      {
         bantu = head;
         while (bantu->next->next!=NULL)
         {
            bantu = bantu->next;
         }
         hapus = bantu->next;
         tmp = hapus->nilai;
         bantu->next=NULL;
         delete hapus;
      }
      else
      {
         tmp=head->nilai;
         head=NULL;
      }
      cout <<  tmp <<  "Terhapus" << endl;
   }
   else cout << "Masih Kosong" << endl;
}
//hapus data tertentu
void hapusDataTertentu(int data){
   Data *hapus, *bantu;
   int tmp;
   if (!isEmpty())
```

```
    {
      if (head->next!=NULL)
      {
         bantu = head;
         while (bantu->next->next!=NULL)
         {
            bantu = bantu->next;
         }
         hapus = bantu->next;
         tmp = hapus->nilai;
         bantu->next=NULL;
         delete hapus;
      }
      else
      {
         tmp=head->nilai;
         head=NULL;
      }
      cout <<  tmp <<  "Terhapus" << endl;
   }
   else cout << "Masih Kosong" << endl;
}

void Cetak ()
{
   if (!isEmpty())
   {
      Data *bantu;
      bantu=head;
      do
      {
         cout << bantu->nilai<< " ";
         bantu=bantu->next;
      }
      while (bantu!=NULL);
      cout << endl;
   }
}


int panjang ()
{
   int count=0;
   if (!isEmpty())
   {
      count=1;
      Data *bantu;
      bantu=head;
      if (bantu->next==NULL)
      {
         count=1;
      }
      else
      {
         do
         {
            count++;
            bantu=bantu->next;
         }
         while (bantu->next!=NULL);
      }
   }
   else
   {
      count=0;
   }
   return count;
}

int main()
{
   awal();
   int pilih, DataBaru;
   do
   {
      system("cls");
      cout << "1. Tambah Depan" << endl;
      cout << "2. Tambah Belakang" << endl;
      cout << "3. Hapus Depan" << endl;
      cout << "4. Hapus Belakang" << endl;
      cout<<"5. Hapus Data Tertentu"<<endl;
      cout << "6. Cetak" << endl;
      cout << "7. Panjang" << endl;
      cout << "8. Keluar" << endl;
      cout << "Pilih : ";
      cin >> pilih;
      switch (pilih)
      {
      case 1:
         cout << "Masukkan Data : ";
         cin >> DataBaru;
         tambahDataDepan(DataBaru);
         break;
}
```

DLL

```
struct Barang{
   int idBarang;
   string namaBarang;
   int stok;
   long harga;
   Barang *prev;
   Barang *next;
};

Barang *newData, *head = NULL, *tail = NULL,
*current;

bool isEmpty(){
   if(head == NULL){
      return true;
   }
   else{
```

```
        return false;
    }
}

bool isSingleNode(){
    if(head->next == NULL){
        return true;
    }
    else{
        return false;
    }
}

void cetak(){
    if(isEmpty()){
        cout<<"List masih kosong"<<endl;
    }
    else{
        system("CLS");
        cout<<"Data Barang :"<<endl;
        cout<<"-------------"<<endl;
        current  = head;
        int i = 1;
        while(current!=NULL){
            cout<<"Barang ke-"<<i<<endl;
            cout<<"Id Barang: "<<current->idBarang<<endl;
            cout<<"Nama Barang: "<<current->namaBarang<<endl;
            cout<<"Stok: "<<current->stok<<endl;
            cout<<"Harga: Rp "<<current->harga<<",00"<<endl;

            current = current->next;
            i++;
        }
    }
}

void buatList(Barang *dataBarang){
    head = dataBarang;
    tail = dataBarang;
}

int panjang ()
{
    int count=0;
    if (!isEmpty())
    {
        count=1;
        Barang *bantu;
        bantu=head;
        if (bantu->next==NULL)
        {
            count=1;
        }
        else
        {
            do
            {
                count++;
                bantu=bantu->next;
            }
            while (bantu->next!=NULL);
        }
    }
    else
    {
        count=0;
    }
    return count;
}

void tambahDepan(Barang *dataBarang){
    if(isEmpty()){
        buatList(dataBarang);
    }
    else{

        dataBarang->next = head;
        head->prev = dataBarang;
        head = dataBarang;
    }
}

void tambahBelakang(Barang *dataBarang){
    if(isEmpty()){
        buatList(dataBarang);
    }
    else{
        tail->next = dataBarang;
        dataBarang->prev = tail;
        tail = dataBarang;
    }
}

void tambahTengah(int pos, Barang *dataBarang){
    if(isEmpty()){
        buatList(dataBarang);
    }

    else if(panjang()<pos){
        tambahBelakang(dataBarang);
    }

    else {
        Barang *nextCurrent;
        current = head;
        int i = 1;
        while(i < pos - 1){
            current = current->next;
            i++;
```

```
        }


        nextCurrent = current->next;
        dataBarang->next = nextCurrent;
        current->next = dataBarang;
        dataBarang->prev = current;
        nextCurrent->prev = dataBarang;
    }
}

void hapusDepan(){
    if(isEmpty()){
        cout<<"Tidak dapat menghapus head karena
list masih kosong"<<endl;
    }
    else{
        if(isSingleNode()){
          head = NULL;
        }
        else{
            Barang *hapus;
            hapus = head;
            head = hapus->next;
            head->prev = NULL;
            hapus->next = NULL;
            delete hapus;
        }
        cout<<"Berhasil menghapus data paling
depan"<<endl;
    }
}

void hapusBelakang(){
    if(isEmpty()){
        cout<<"Tidak dapat menghapus tail karena list
masih kosong"<<endl;
    }
    else{
        if(isSingleNode()){
            head = NULL;
        }
        else{
            Barang *hapus;
            hapus = tail;
            tail = hapus->prev;
            tail->next = NULL;
            delete hapus;
        }
        cout<<"Berhasil menghapus data paling
belakang"<<endl;

    }
}

void hapusTengah(int pos){
    if(isSingleNode()){
        head = NULL;
        cout<<"Berhasil hapus node"<<endl;
    }

    else if(panjang()<pos){
        hapusBelakang();
    }
    else
    {
        Barang *hapus;
        current = head;
        int i = 1;
        while(i < pos){
            current = current->next;
            i++;
        }
        hapus = current->next;
        current->next = hapus->next;
        delete hapus;
        cout<<"Berhasil hapus node di posisi
"<<pos<<endl;
    }
}

void search(int data)
{
    Barang *cari = new Barang;
    cari = head;
    if (head == NULL)
    {
        cout << "data kosong" << endl;
    }
    else
    {
        while (cari->idBarang != data)
        {
            cari = cari->next;
        }
        cout << cari->namaBarang;
        cout<<"  ditemukan" << endl;
    }
}

Barang *inputData()
{
    newData = new Barang();
    cout<<"Id barang: ";
    cin>>newData->idBarang;
    cout<<"Nama barang: ";
    getline(cin>>ws, newData->namaBarang);
    cout<<"Stok: ";
    cin>>newData->stok;
    cout<<"Harga: Rp ";
    cin>>newData->harga;
    newData->next = NULL;
```

```
    return newData;
}

void aksi(int pilih){
    Barang *inputan;
    int posisi;
    switch(pilih){
        case 0 : cout<<"Terima kasih"<<endl;exit(0);
        case 1 : system("CLS");
                 cout<<"Tambah data di depan"<<endl;
                 inputan = inputData();
                 tambahDepan(inputan);break;
        case 2 : system("CLS");
                 cout<<"Tambah data di
belakang"<<endl;
                 inputan = inputData();
                 tambahBelakang(inputan);break;
        case 3 : cetak();break;
        case 4 : hapusDepan();break;
        case 5 : hapusBelakang();break;
        case 6 : cout<<"Masukkan posisi:
";cin>>posisi;
                 inputan = inputData();
                 tambahTengah(posisi, inputan);break;
        case 7 : cout<<"Masukkan posisi yang mau
dihapus: ";cin>>posisi;
                 hapusTengah(posisi);break;
        case 8 : cout<<"Panjang list:
"<<panjang()<<endl;break;
        default: cout<<"Pilihan "<<pilih<<" belum
tersedia"<<endl;
                 exit(0);break;
    }
}
```

TREE

```
struct Tree
{
  char huruf;
  Tree *left;
  Tree *right;
};

Tree *node, *nodeBaru, *root = NULL, *current;

Tree *tambahNode(Tree *&current, char data)
{
  if (current == NULL)
  {
    current = new Tree;
    current->huruf = data;
    current->left = NULL;
    current->right = NULL;
  }
  else
  {
    if (data < current->huruf)
    {
      tambahNode(current->left, data);
    }
    else
    {
      tambahNode(current->right, data);
    }
  }

  return current;
}

void preOrder(Tree *current)
{
  if (current != NULL)
  {
    cout << " " << current->huruf;
    preOrder(current->left);
    preOrder(current->right);
  }
}

void inOrder(Tree *current)
{
  // kiri parent kanan
  if (current != NULL)
  {
    inOrder(current->left);
    cout << " " << current->huruf;
    inOrder(current->right);
  }
}

void postOrder(Tree *current)
{
  // kiri kanan parent
  if (current != NULL)
  {
    postOrder(current->left);
    postOrder(current->right);
    cout << " " << current->huruf;
  }
}

void menu(int pilih)
{
  char data;
  Tree *inputan;
  switch (pilih)
  {
  case 1:
    cout << "Masukkan Satu Huruf : ";
    cin >> data;
    inputan = tambahNode(node, data);
    break;
  case 2:
```

```
    cout << "Masukkan Satu Huruf : ";
    cin >> data;
    inputan = tambahNode(node, data);
    break;
  case 3:
    preOrder(node);
    cout << "\n";
    break;
  case 4:
    inOrder(node);
    cout << "\n";
    break;
  case 5:
    postOrder(node);
    cout << "\n";
    break;
  case 0:
    exit(0);
    break;

  default:
    cout << "Masukkan pilihan yang tepat" << endl;
    break;
  }
}

int main()
{
  int pilih;
  do
  {
    cout << "1. Tambah Kiri" << endl;
    cout << "2. Tambah Kanan" << endl;
    cout << "3. Pre Order" << endl;
    cout << "4. In Order" << endl;
    cout << "5. Post Order" << endl;
    cout << "0. Exit" << endl;
    cout << "Masukkan Pilihan Menu : ";
    cin >> pilih;
    menu(pilih);
  } while (pilih != 0);
}
```