

LAPORAN PRAKTIKUM
SESI 8
PRAKTIKUM COMP6362 – DATA STRUCTURES
KELAS BE20



Oleh :
2440008600 – Andru Baskara Putra

SEMESTER GENAP 2020/2021
BINA NUSANTARA UNIVERSITY
MALANG

A. Kode Program

Source Code

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<conio.h>
#include<malloc.h>

struct tree{
    char name[200];
    int num;
    struct tree *kanan, *kiri;
};
typedef struct tree node;

node*buat(char nama[],int nomor){
    node*nodebaru = (node *)malloc(sizeof(node));
    strcpy(nodebaru->name,nama);
    nodebaru->num = nomor;
    nodebaru->kiri = NULL;
    nodebaru->kanan = NULL;
    return nodebaru;
}

void tambah(node **temp, char nama[], int nomor,int height){
    if (height<4)
    {
        if (*temp==NULL)
        {
            *temp = buat(nama,nomor);
            (*temp)->kiri=NULL;
            (*temp)->kanan=NULL;
            printf("\n Berhasil! \n");
        }else
        {
            char pos[200];
            do
            {
                printf("\nMau tambah di kiri atau kanan nya si %s?",(*temp)->name);
                scanf("%s",pos);
            } while (strcmp(pos,"kiri")!=0 && strcmp(pos,"kanan")!=0);

            if (strcmp(pos,"kiri")==0)
            {
                tambah(&(*temp)->kiri,nama,nomor,height+1);
            }else
```

```

    {
        tambah(&>(*temp)->kanan,nama,nomor,height+1);
    }

}

}else
{
    printf("\nGabisa dong bund, melebihi maksimal\n");
}

}

node *cari(node *root, int nomor){
    if(root!=NULL){
        if(root->num==nomor) {
            return root;
        }
        else{
            node *temp;
            temp=cari(root->kiri,nomor);
            if(temp==NULL){
                temp=cari(root->kanan,nomor);
            }
            return temp;
        }
    }
    else{
        return 0;
    }
}

void showlist(node*root){
    if (root==NULL)
    {
        return;
    }else
    {
        printf("%-20s (%d)\n", root->name, root->num);
    }
}

```

```

        showlist(root->kiri);
        showlist(root->kanan);
    }

}

void preorder(node *pohon){
    if (pohon==NULL)
    {
        return;
    }else
    {
        printf(" %d\n",pohon->num);
        preorder(pohon->kiri);
        preorder(pohon->kanan);
    }
}

void inorder(node *pohon){
    if (pohon==NULL)
    {
        return;
    }else
    {
        inorder(pohon->kiri);
        printf(" %d\n",pohon->num);
        inorder(pohon->kanan);
    }
}

void postorder(node *pohon){
    if (pohon==NULL)
    {
        return;
    }else
    {
        postorder(pohon->kiri);
        postorder(pohon->kanan);
        printf(" %d\n",pohon->num);
    }
}

```

```

node *hapusSemua(node *pohon){
    if (pohon!=NULL)
    {
        hapusSemua(pohon->kiri);
        hapusSemua(pohon->kanan);
        free(pohon);
    }
    return 0;
}

void hapus(node **temp, int nomor){
    if ((*temp)!=NULL && nomor==( *temp)->num)
    {
        hapusSemua(*temp);
        (*temp)=NULL;
    }else if ((*temp)!=NULL)
    {
        hapus(&(*temp)->kiri,nomor);
        hapus(&(*temp)->kanan,nomor);
    }
}

int main(){
    int opsi=0;
    char nama[200];
    int nomor;

    node *root=NULL; node *res;

    while (opsi < 5 ) {
        printf("=====\n\n");
        printf("          DAFTAR PEMAIN          \n\n");
        printf("=====\n");
        printf("1. Tampilkan Pemain\n");
        printf("2. Tambah Pemain\n");
        printf("3. Hapus Pemain\n");
        printf("4. Inorder, Preorder, Postorder\n");
        printf("5. Kill Program\n");
        scanf("%d", &opsi);
        if(opsi==1){
            if(!root){
                printf("\nGaada siapa2 bund\n");
            }else{
                showlist(root);
            }
        }
    }
}

```

```
}else if(opsi==2){
    do{
        printf("Tulis Nama Pemain: ");
        scanf("%s",nama);
    }while(strlen(nama)<3 || strlen(nama)>20);
    while(1){
        printf("Tulis Nomor: ");
        scanf("%d", &nomor);
        if(nomor>0 && nomor<100){
            res = cari(root,nomor);
            if(res){
                printf("\nUdah ada bund :( cari nomor lain\n\n");
            }else{
                break;
            }
        }else{
            printf("Nomor nya jangan di bawah 0 bund, juga jangan lebih dari 100\n");
        }
    }
    tambah(&root,nama,nomor,0);
}

}else if(opsi==3){
    if(!root){
        printf("\nGaada nama kayak gitu bund\n");
    }else{
        while(1){
            showlist(root);
            printf("\nInput Nilai: ");
            scanf("%d",&nomor);
            if(nomor>0 && nomor<100){
                node *temp;
                res = cari(root,nomor);
                if(!res){
                    printf("\n\nGaada nama kayak gitu bund\n");
                }else{
                    hapus(&root,nomor);
                    printf("\n\nBerhasil dihapus\n");
                    break;
                }
            }
        }else{
            printf("Nomor nya jangan di bawah 0 bund, juga jangan lebih dari 100\n");
        }
    }
}

}
```

```
    if(!root){
        printf("\nMasih kosongan \n");
    }else{
        printf("\n Inorder :\n");
        inorder(root);
        printf("\n Preorder :\n");
        preorder(root);
        printf("\n Postorder :\n");
        postorder(root);
    }
}
}
exit(0);
}
```

B. Deskripsi Program

1. Struct tree

Berisikan kumpulan data-data yang akan digunakan dalam program nanti

2. Node*buat()

Merupakan fungsi dimana program akan melakukan proses pembentukan data

3. Void tambah()

Merupakan fungsi dimana program akan menambah data dibagian root, kiri, maupun kanan rangkaian tree

4. Node *cari ()

Merupakan fungsi dimana program akan mencari data yang tersedia dalam tree sesuai keinginan user

5. Void preorder()

Merupakan fungsi dimana program akan menampilkan data berdasarkan preorder traversal

6. Void inorder()

Merupakan fungsi dimana program akan menampilkan data berdasarkan inorder traversal

7. Void postorder()

Merupakan fungsi dimana program akan menampilkan data berdasarkan postorder traversal

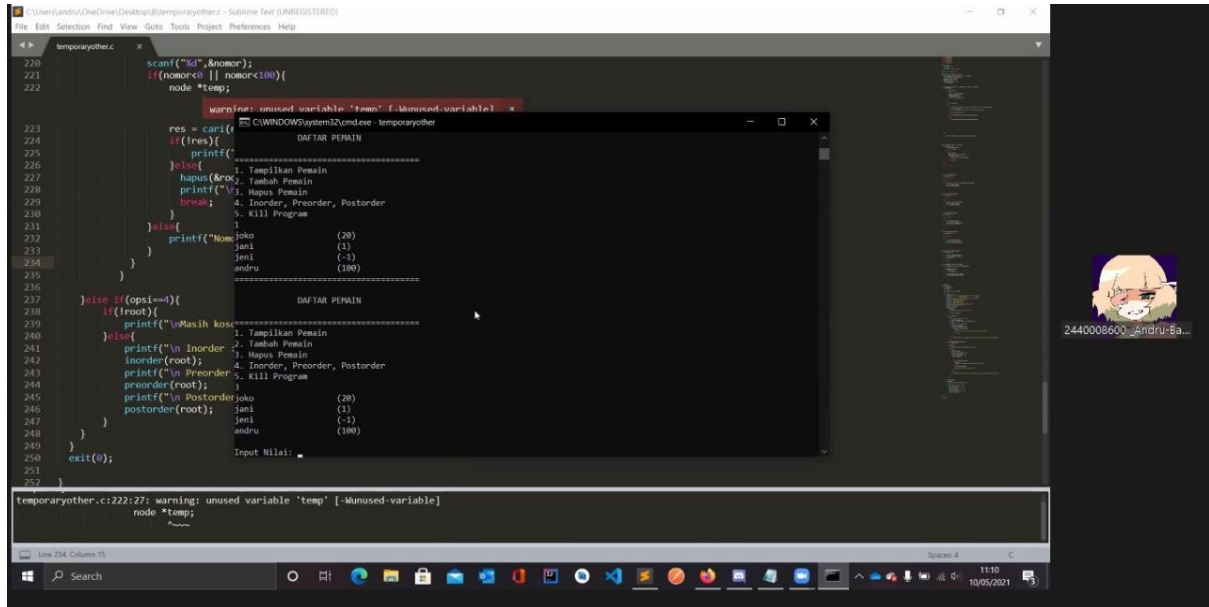
8. Void showlist()

Fungsi ini akan memunculkan data-data yang telah dibuat ke layar

9. int main()

Berisikan proses utama dari program, user akan diperlihatkan beberapa pilihan untuk menambah data, menghapus data, mengurut data dan juga memberhentikan jalan nya program.

Bukti Presentasi



The screenshot shows a C++ program in Sublime Text. A warning message is displayed: "warning: unused variable 'temp' [-Wunused-variable]". The code is as follows:

```
220 scanf("%d",&nomor);
221 if(nomor<0 || nomor>100){
222     node *temp;
223
224     res = cari(
225         if(!res){
226             printf("
227             1. Tampilkan Pemin
228             hapus(&root);
229             printf("
230             break;
231             4. Inorder, Preorder, Postorder
232             5. Kill Program
233         }
234         printf("
235         1
236         joko (20)
237         jani (1)
238         jani (-1)
239         andru (100)
240     )
241     }
242     }
243     }
244     }
245     }
246     }
247     }
248     }
249     }
250     }
251     }
252     }
```

The warning message is: "warning: unused variable 'temp' [-Wunused-variable]".

The code is a C++ program that uses a linked list to store names and ages. It includes functions for adding, deleting, and displaying nodes. The program uses a variable named 'temp' to store a pointer to a new node, but it is not used in the code.