Exam Data Structure

Format file : [KodeKelas] – [Nama].zip

Berisi file .cpp

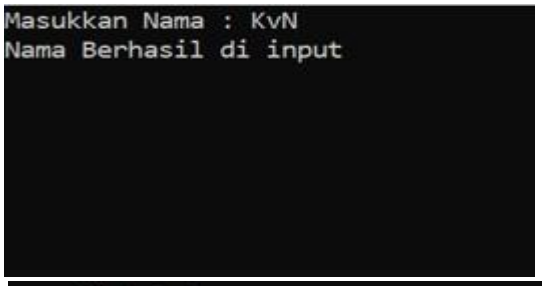
Integrity more valuable than result

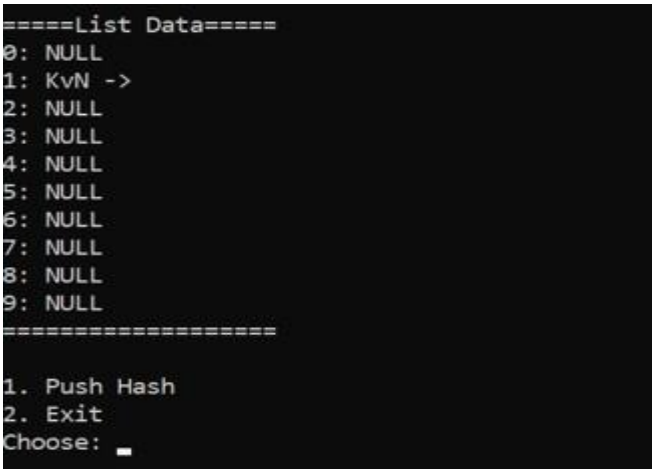
Buatah sebuah aplikasi untuk masing-masing soal menggunakan konsep Data Structure dengan deskripsi sebagai berikut :

I. Bobot nilai 20 poin

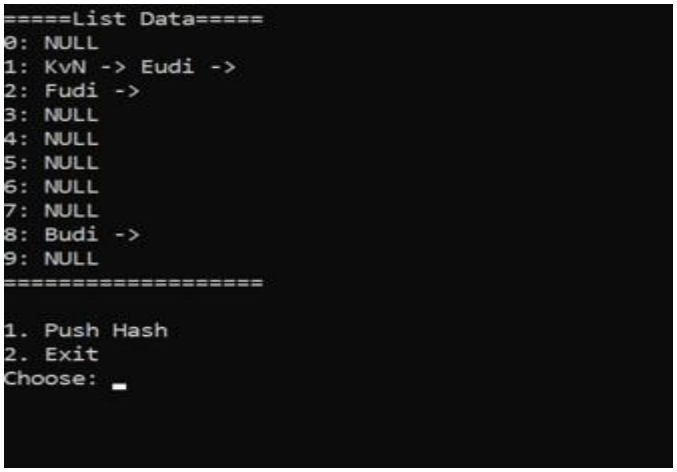
Lengkapi potongan code Division Hash dengan konsep Chaining implementasi double linked list.

- Terdapat alokasi memory sebanyak 10 tempat.
- Nama akan dilakukan Hashing berdasarkan total ASCII karakter lalu dimasukkan ke salah satu memory dengan cara modulus.
  Contoh input : KvN
  K = 75
  v = 118
  N = 78
  Total ASCII KvN : 271.
  271 % 10 = 1, maka KvN dimasukkan ke lokasi memori 1.





- Jika nama yang dilakukan hashing memiliki lokasi memori yang sudah ditempati, maka aplikasi akan menggunakan konsep “CHAINING”, yaitu data akan disambungkan ke data sebelumnya pada lokasi memori tersebut.
  Contoh input dan total ASCI :
  - Budi : 388 % 10 = 8
  - Fudi : 392 % 10 = 2
  - Eudi : 391 % 10 = 1



Berikut potongan code yang harus dilengkapi (file potongan code diberikan coach).

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

struct Node{
	char nama[100];
	struct Node* next,*prev;
}*head[10],*tail[10],*current;

int hash(char inputanNama[]){
	int panjangString = strlen(inputanNama);
	int totalAscii = 0;
	for (int i=0; i<panjangString; i++){
		// code logic hash
	}
	// return;
}

void pushHash(char inputanNama[]){
	// code push untuk hash
}
```

```
void viewAll(){
	for (int i=0; i<10; i++){
		printf("%d: ",i);
		if (head[i] == NULL){
			printf("NULL\n");
		}
		else{
			current = head[i];
			while (current!=NULL){
				printf("%s -> ",current->nama);
				current = current->next;
			}
			puts("");
		}
	}
}
int main(){
	int pilih;
	char nama[255];
	do{

		system("CLS");
		printf("=====List Data=====\n");
		viewAll();
		printf("====================\n\n");
		printf("1. Push Hash\n");
		printf("2. Exit\n");
		printf("Choose: ");
		scanf("%d" , &pilih);fflush(stdin);

		switch(pilih){
			case 1:
				printf("Masukkan Nama : ");
				scanf("%[^\n]" , nama);fflush(stdin);
				pushHash(nama);
				printf("Nama Berhasil di input");getchar();
				break;
		}
	}while(pilih != 2);
	return 0;
}
```

II. Bobot nilai 20 poin

Lengkapi potongan code single linked list menggunakan konsep **Queue** berikut:

- Diberikan code untuk push data.

```
==========
Queue List
==========

Andrianus(18) ->

1. Push Queue
2. Pop Queue
3. Exit
Choose : 1
Masukkan Nama : Bandre
Masukkan Umur : 19
```

- Buatlah code untuk Pop dengan konsep Queue beserta validasi.
  Jika tidak ada data dalam list, maka menu pop akan memberikan pesan "Tidak ada data!". Sebaliknya jika data yang akan di-pop tersedia, maka aplikasi akan menghapus data dari list dan menampilkan pesan "Data berhasil dihapus".

  Data berhasil dihapus:

```
==========
Queue List
==========

Bandre(19) -> Andrianus(18) ->

1. Push Queue
2. Pop Queue
3. Exit
Choose : 2
Data Berhasil di hapus
```

```
==========
Queue List
==========

Bandre(19) ->

1. Push Queue
2. Pop Queue
3. Exit
Choose :
```

  Gambar jika data kosong :

```
==========
Queue List
==========

No Data!

1. Push Queue
2. Pop Queue
3. Exit
Choose : 2
Tidak ada data !
```

Berikut potongan code yang harus dilengkapi (file potongan code diberikan coach).

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

struct Mahasiswa{
	char nama[255];
	int umur;
	Mahasiswa *next;
}*head , *tail , *curr;

void push(char nama[] , int umur){
	curr = (Mahasiswa*)malloc(sizeof(Mahasiswa));
	strcpy(curr->nama,nama);
	curr->umur = umur;
	curr->next = NULL;

	if(head == NULL){
		head=tail=curr;
	}else{
		curr->next = head;
		head = curr;
	}
}

void pop(){
	// code pop dari queue
}

void view(){
	curr=head;
	if(head == NULL){
		printf("No Data!");
	}else{
		while(curr != NULL){
			printf("%s(%d) -> ", curr->nama , curr->umur);
			curr=curr->next;
		}
	}
}
```

```
int main(){
	int pilih , umur;
	char nama[255];

	do{
		system("CLS");
		printf("==========\n");
		printf("Queue List\n");
		printf("==========\n\n");
		view();
		printf("\n\n1. Push Queue\n");
		printf("2. Pop Queue\n");
		printf("3. Exit\n");
		printf("Choose : ");
		scanf("%d" , &pilih);fflush(stdin);

		switch(pilih){
			case 1:
				printf("Masukkan Nama : ");
				scanf("%[^\n]" , nama);fflush(stdin);
				printf("Masukkan Umur : ");
				scanf("%d" , &umur);fflush(stdin);
				push(nama,umur);
				break;

			case 2:
				pop();
				break;
		}
	}while(pilih != 3);
	return 0;
}
```

III. Bobot nilai 60 poin

(10 point) Buatlah aplikasi untuk Binary Tree yang memiliki menu berikut beserta validasi:

- Input menu hanya antara 1-5. Jika tidak aplikasi akan kembali meminta user untuk meminta input menu.
- Ketika user memilih menu 1-4, aplikasi akan menjalankan menu tersebut dan kembali ke home.

```
Binary Tree
1. Push data
2. View Preorder
3. View Inorder
4. View Postorder
5. Exit
Choose :
```

(20 point) Menu 1

Push data ke dalam Binary Tree dengan ketentuan sebagai berikut:

- Jika root masih kosong, maka angka yang diinput sebagai root.
- Jika angka lebih kecil, maka akan ditempatkan di left child.
- Jika angka lebih besar, maka akan ditempatkan di right child.

Contoh input: 50, 25, 75, 10, 30, 60, 80.

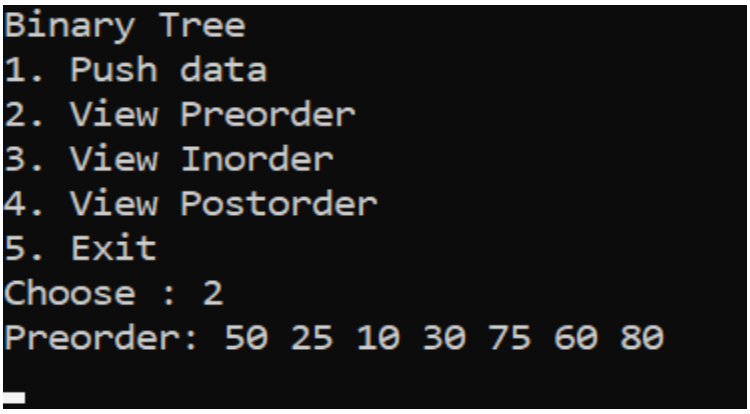
```
Binary Tree
1. Push data
2. View Preorder
3. View Inorder
4. View Postorder
5. Exit
Choose : 1
Masukkan Angka : 50
Data berhasil di input
```

Jika menginput angka yang sudah ada, maka akan menampilkan pesan “Insert gagal, data sudah ada”.

```
Binary Tree
1. Push data
2. View Preorder
3. View Inorder
4. View Postorder
5. Exit
Choose : 1
Masukkan Angka : 50
Insert gagal, data sudah ada
```

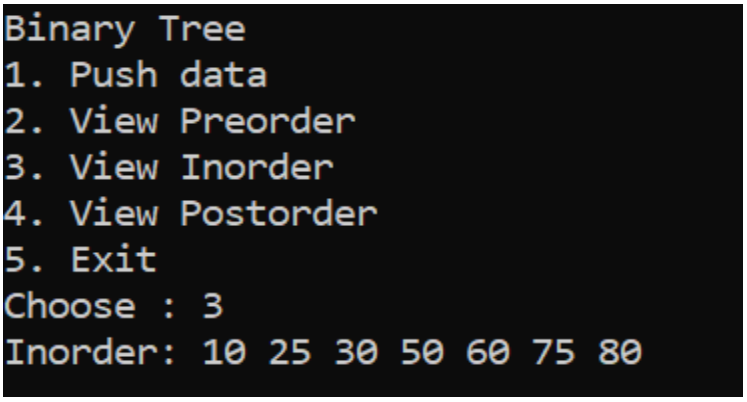
(10 point) Menu 2

Menampilkan data Binary Tree dalam bentuk Preorder.



(10 point) Menu 3

Menampilkan data Binary Tree dalam bentuk Inorder.



(10 point) Menu 4

Menampilkan data Binary Tree dalam bentuk Postorder.