Neuroscience

Neha KV

February 26, 2024

1 Problem Statement

C program to search for a neuron using linked list.

2 Approach Used

I used the singly linked list approach for searching the details of neuron to know where exactly a particular neuron lies

3 Functions Used

3.1 Neuron search function

```
void search(NODE first)
{
    NODE cur=first;
    int searchid;
    printf("Enter the neuron ID to be searched:");
    scanf("%d",&searchid);
    while(cur!=NULL)
    {
        if(searchid==cur->neuronID)
        {
            printf("Match Found. The details of neuron are\n");
            putdetails(cur);
            return;
        }
        cur=cur->link;
    }
    printf("neuron not found\n");
}
```

4 Final Code

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

// Define Node structure for linked list
struct node {
    int neuronID;
    char neuronType[20];
```

```
    char neuronLocation[50];
    char neuronFunction[100];
    struct node * link;
};
typedef struct node * NODE;
NODE getnode()
{
    NODE x;
    x=(NODE)malloc(sizeof(struct node));
    if(x==NULL)
    {
        printf("Memomry not allocated");
        exit(0);
    }
    return x;
}
void getdetails(NODE temp)
{
    printf("NeuronID:");
    scanf("%d",&(temp->neuronID));
    printf("Neuron Type:");
    scanf("%s",(temp->neuronType));
    printf("Neuron Location:");
    scanf("%s",(temp->neuronLocation));
    printf("Neuron Function:");
    scanf("%s",(temp->neuronFunction));
}
NODE insert_info(NODE first)
{
    NODE cur,prev,temp;
    temp=getnode();
    getdetails(temp);
    temp->link=NULL;
    if(first==NULL)
    {
        return temp;
    }
    cur=first;
    prev=NULL;
    while(cur!=NULL)
    {
        prev=cur;
        cur=cur->link;
    }
    prev->link=temp;
    return first;
}
void putdetails(NODE temp)
{
    printf("Neuron ID:%d\n", temp->neuronID);
        printf("--Type:%s\n", temp->neuronType);
        printf("--Location:%s\n", temp->neuronLocation);
        printf("--Function:%s\n", temp->neuronFunction);
}
// Function to display all nodes in the linked list
void display(NODE first)
```

```
{
    NODE cur = first;
    if(cur==NULL)
    {
        printf("Empty List\n");
        return;
    }
    while (cur != NULL)
    {
        printf("\n");
        putdetails(cur);
        cur = cur->link;
    }
}
void search(NODE first)
{
    NODE cur=first;
    int searchid;
    printf("Enter the neuron ID to be searched:");
    scanf("%d",&searchid);
    while(cur!=NULL)
    {
        if(searchid==cur->neuronID)
        {
            printf("Match Found.The details of neuron are\n");
            putdetails(cur);
            return;
        }
        cur=cur->link;
    }
    printf("neuron not found\n");
}
void free_list(NODE first) {
    NODE temp;
    while (first != NULL) {
        temp = first;
        first = first->link;
        free(temp);
    }
}

int main()
{
    NODE first=NULL;
    int ch;
    while(1)
    {
        printf("Enter the choice\n1.insert info\n2.display\n3.search\n4.break\n");
        scanf("%d",&ch);
        switch(ch)
        {
            case 1:first=insert_info(first);
            break;
            case 2:display(first);
            break;
            case 3:search(first);
```

```
            break;
            case 4:
                free_list(first);
                exit(0);
            default:
                printf("Invalid choice. Please enter a valid option.\n");
        }
    }
}
```

5 Output Screenshots

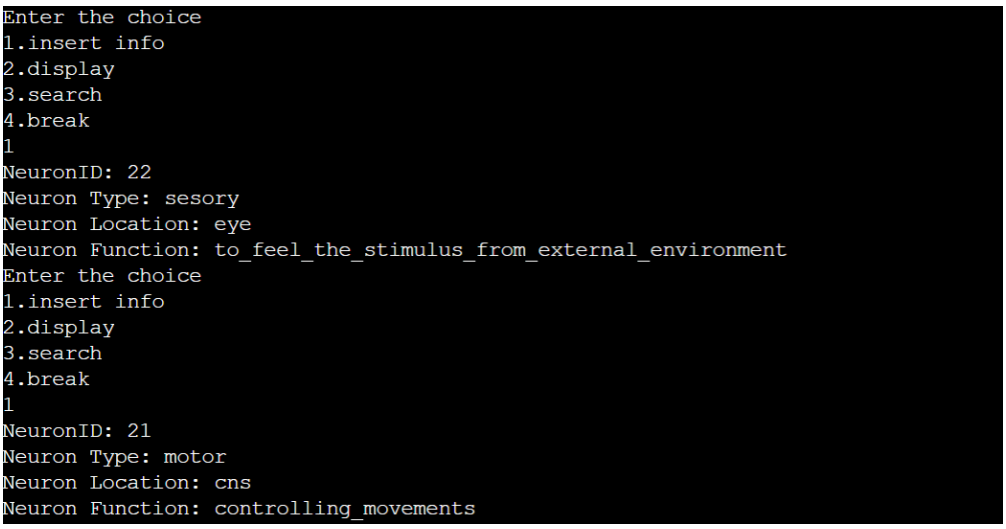

Figure 1: Output for Test Case

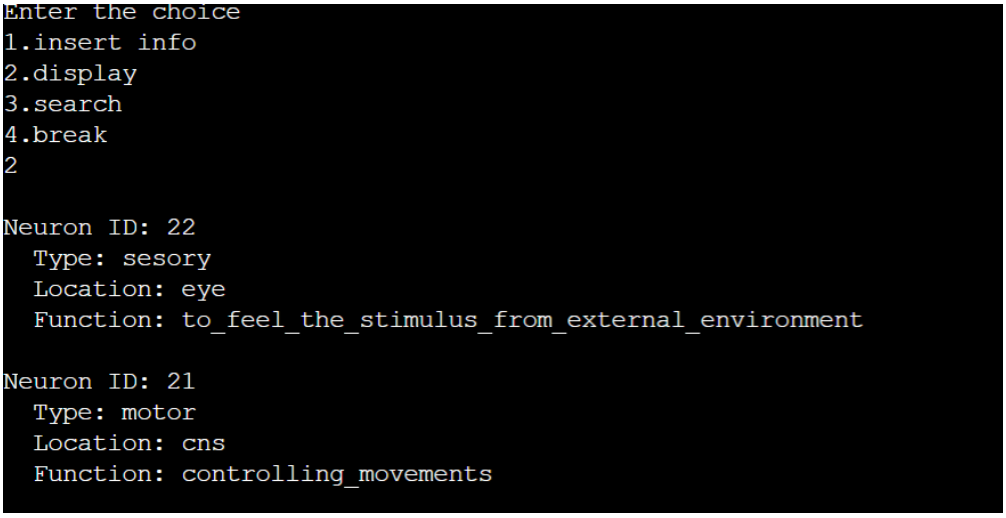

Figure 2: Output for Test Case

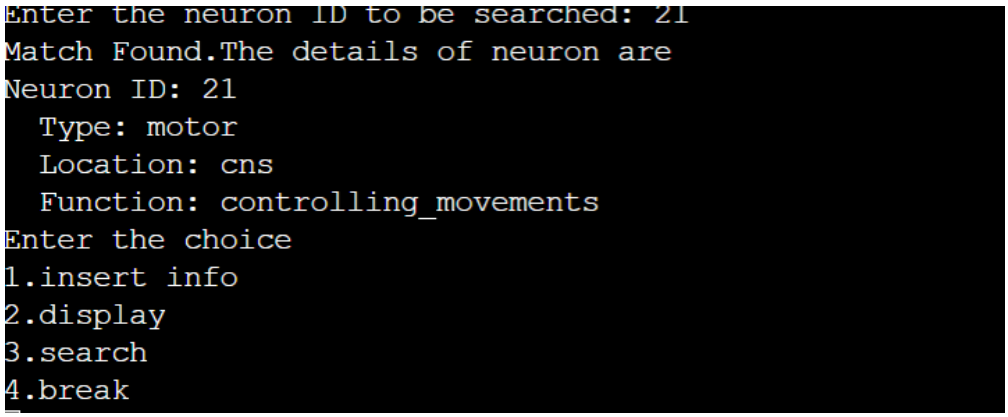

Figure 3: Output for Test Case