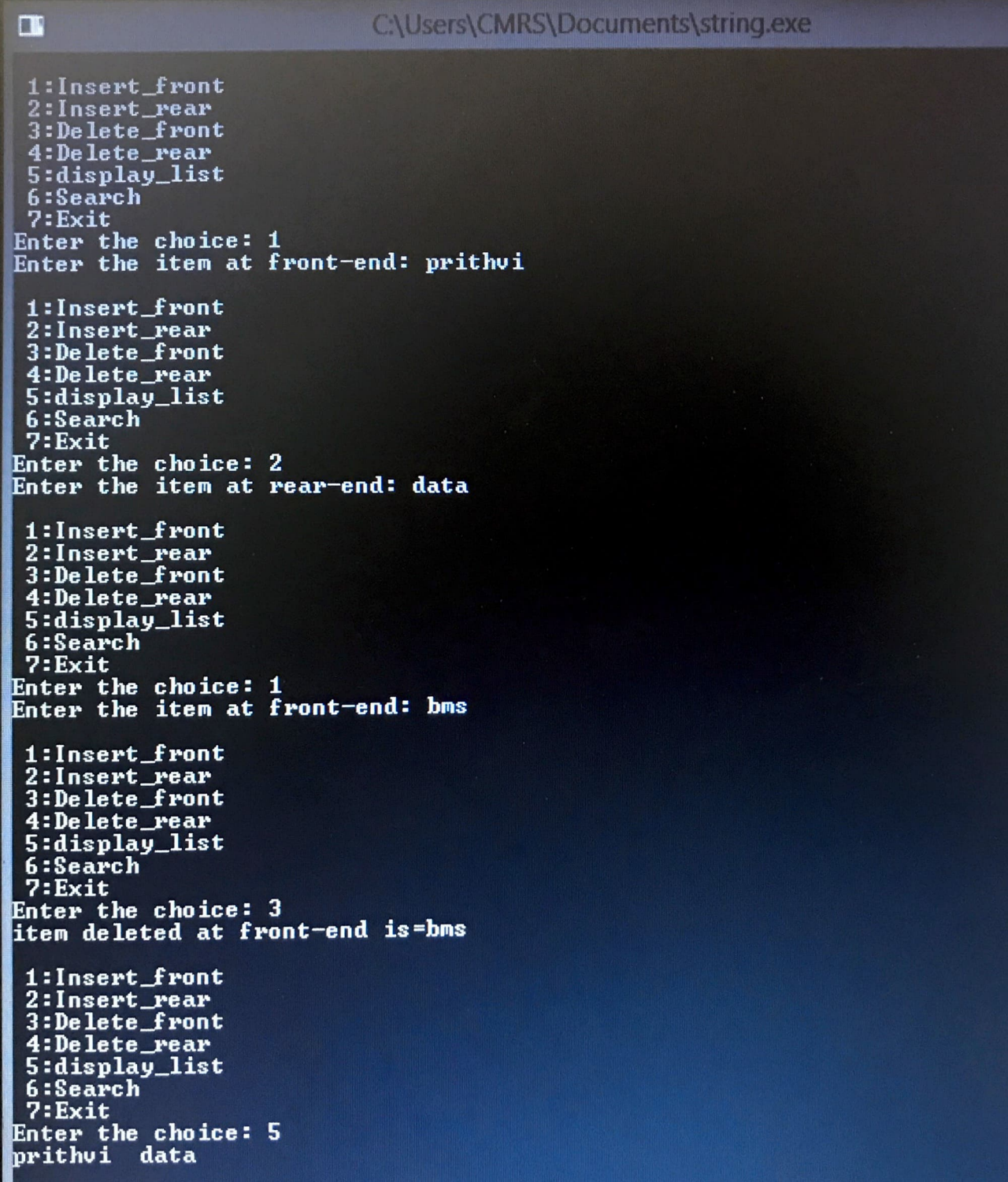
```c
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<string.h>
struct node
{
char name[20];
struct node *link;
};
typedef struct node *NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
{
printf("Memory full\n");
exit(0);
}
return x;
}
void freenode(NODE x)
{
free(x);
}

NODE insert_front(NODE first,char item[])
{
NODE temp;
temp=getnode();
strcpy(temp->name,item);
temp->link=NULL;
if(first==NULL)
return temp;
temp->link=first;
first=temp;
```

```
37    return first;
38    }
39
40    NODE insert_rear(NODE first,char item[])
41    {
42    NODE temp,cur;
43    temp=getnode();
44    strcpy(temp->name,item);
45    temp->link=NULL;
46    if(first==NULL)
47    return temp;
48    cur=first;
49    while(cur->link!=NULL)
50    cur=cur->link;
51    cur->link=temp;
52    return first;
53    }
54
55
56    NODE delete_front(NODE first)
57    {
58    NODE temp;
59    if(first==NULL)
60    {
61    printf("list is empty cannot delete\n");
62    return first;
63    }
64    temp=first;
65    temp=temp->link;
66    printf("item deleted at front-end is=%s\n",first->name);
67    free(first);
68    return temp;
69    }
70
71    NODE delete_rear(NODE first)
72    {
```

```
73      NODE cur,prev;
74      if(first==NULL)
75      {
76      printf("list is empty cannot delete\n");
77      return first;
78      }
79      if(first->link==NULL)
80      {
81      printf("Item deleted is %s\n",first->name);
82      free(first);
83      return NULL;
84      }
85      prev=NULL;
86      cur=first;
87      while(cur->link!=NULL)
88      {
89      prev=cur;
90      cur=cur->link;
91      }
92      printf("item deleted at rear-end is %s",cur->name);
93      free(cur);
94      prev->link=NULL;
95      return first;
96      }
97      void search(char key[],NODE first)
98      {
99          NODE cur;
100         if(first==NULL)
101         {
102             printf("List is empty\n");
103             return;
104         }
105         cur=first;
106         while(cur!=NULL)
107         {
108             if(strcasecmp(key,cur->name)==0)
```

```
108             if(strcasecmp(key,cur->name)==0)
109                 break;
110             cur = cur->link;
111         }
112         if(cur==NULL)
113         {
114             printf("Name not found\n");
115             return;
116         }
117             printf("Name found\n");
118     }
119     void display(NODE first)
120     {
121      NODE temp;
122      if(first==NULL)
123      printf("List empty cannot display items\n");
124      for(temp=first;temp!=NULL;temp=temp->link)
125       {
126        printf("%s  ",temp->name);
127       }
128        printf("\n");
129     }
130
131     int main()
132     {
133     int choice,pos;
134     char item[10];
135     NODE first=NULL;
136
137     for(;;)
138     {
139     printf("\n 1:Insert_front \n 2:Insert_rear \n 3:Delete_front \n 4:Delete_rear \n 5:display_list \n 6:Search \n 7:Exit\n");
140     printf("Enter the choice: ");
141     scanf("%d",&choice);
142     switch(choice)
143      {
```

```
134  char item[10];
135  NODE first=NULL;
136
137  for(;;)
138  {
139  printf("\n 1:Insert_front \n 2:Insert_rear \n 3:Delete_front \n 4:Delete_rear \n 5:display_list \n 6:Search \n 7:Exit\n");
140  printf("Enter the choice: ");
141  scanf("%d",&choice);
142  switch(choice)
143   {
144    case 1:printf("Enter the item at front-end: ");
145       scanf("%s",&item);
146       first=insert_front(first,item);
147       break;
148    case 2:printf("Enter the item at rear-end: ");
149       scanf("%s",&item);
150       first=insert_rear(first,item);
151       break;
152    case 3:first=delete_front(first);
153       break;
154    case 4:first=delete_rear(first);
155       break;
156    case 5:display(first);
157       break;
158    case 6:printf("Enter the name to be searched: ");
159       scanf("%s",&item);
160       search(item,first);
161       break;
162   default:exit(0);
163       break;
164   }
165  }
166
167  }
168
```


C:\Users\CMRS\Documents\string.exe
1:Insert_front
2:Insert_rear
3:Delete_front
4:Delete_rear
5:display_list
6:Search
7:Exit
Enter the choice: 1
Enter the item at front-end: prithvi
1:Insert_front
2:Insert_rear
3:Delete_front
4:Delete_rear
5:display_list
6:Search
7:Exit
Enter the choice: 2
Enter the item at rear-end: data
1:Insert_front
2:Insert_rear
3:Delete_front
4:Delete_rear
5:display_list
6:Search
7:Exit
Enter the choice: 1
Enter the item at front-end: bms
1:Insert_front
2:Insert_rear
3:Delete_front
4:Delete_rear
5:display_list
6:Search
7:Exit
Enter the choice: 3
item deleted at front-end is=bms
1:Insert_front
2:Insert_rear
3:Delete_front
4:Delete_rear
5:display_list
6:Search
7:Exit
Enter the choice: 5
prithvi data