**ASSIGNEMENT 8**

Programme:

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

#include<string.h>

#define COUNT 15

struct node

{

int ram,rom,price;

float fcam,bcam,disp;

char brand[50],model[50],connectivity[50],processor[50],os[50];

struct node \*lchild;

struct node \*rchild;

}\*root;

typedef struct node1

{

int ram,rom,price;

float fcam,bcam,disp;

char brand[50],model[50],connectivity[50],processor[50],os[50];

struct node1 \*next;

}node1;

node1\* insert\_loc(node1\*,node1\*);

void create\_queue();

void del\_queue();

void display\_queue(node1 \*front);

node1 \*rear=NULL;

node1 \*front=NULL;

node1 \*p,\*head;

int i,n;

char ch;

void find(int item,struct node \*\*par,struct node \*\*loc)

{

struct node \*ptr,\*ptrsave;

if(root==NULL) /\*tree empty\*/

{

\*loc=NULL;

\*par=NULL;

return;

}

if(item==root->price) /\*item is at root\*/

{

\*loc=root;

\*par=NULL;

return;

}

/\*Initialize ptr and ptrsave\*/

if(item<root->price)

ptr=root->lchild;

else

ptr=root->rchild;

ptrsave=root;

while(ptr!=NULL)

{

if(item==ptr->price)

{ \*loc=ptr;

\*par=ptrsave;

return;

}

ptrsave=ptr;

if(item<ptr->price)

ptr=ptr->lchild;

else

ptr=ptr->rchild;

}/\*End of while \*/

\*loc=NULL; /\*item not found\*/

\*par=ptrsave;

}

void insert(int item)

{ struct node \*tmp,\*parent,\*location;

find(item,&parent,&location);

if(location!=NULL)

{

printf("Item already present");

return;

}

tmp=(struct node \*)malloc(sizeof(struct node));

tmp->price=item;

printf("\n Enter brand of mobile:");

scanf("%s",(tmp->brand));

printf("\n Enter model name:");

scanf("%s",(tmp->model));

printf("\n Enter connectivity type:");

scanf("%s",(tmp->connectivity));

printf("\n Enter processor:");

scanf("%s",(tmp->processor));

printf("\n Enter operating system:");

scanf("%s",(tmp->os));

printf("\n Enter size of RAM in GB:");

scanf("%d",&(tmp->ram));

printf("\n Enter size of ROM:");

scanf("%d",&(tmp->rom));

while(!(tmp->rom<512))

printf("\n Enter front camera:");

scanf("%f",&(tmp->fcam));

printf("\n Enter back camera:");

scanf("%f",&(tmp->bcam));

printf("\n Enter size of display:");

scanf("%f",&(tmp->disp));

tmp->lchild=NULL;

tmp->rchild=NULL;

if(parent==NULL)

root=tmp;

else

if(item<parent->price)

parent->lchild=tmp;

else

parent->rchild=tmp;

}/\*End of insert()\*/

void case\_a(struct node \*par,struct node \*loc )

{

if(par==NULL) /\*item to be deleted is root node1\*/

root=NULL;

else

if(loc==par->lchild)

par->lchild=NULL;

else

par->rchild=NULL;

}/\*End of case\_a()\*/

void case\_b(struct node \*par,struct node \*loc)

{

struct node \*child;

/\*Initialize child\*/

if(loc->lchild!=NULL) /\*item to be deleted has lchild \*/

child=loc->lchild;

else /\*item to be deleted has rchild \*/

child=loc->rchild;

if(par==NULL ) /\*Item to be deleted is root node1\*/

root=child;

else

if( loc==par->lchild) /\*item is lchild of its parent\*/

par->lchild=child;

else /\*item is rchild of its parent\*/

par->rchild=child;

}/\*End of case\_b()\*/

void case\_c(struct node \*par,struct node \*loc)

{

struct node \*ptr,\*ptrsave,\*suc,\*parsuc;

/\*Find inorder successor and its parent\*/

ptrsave=loc;

ptr=loc->rchild;

while(ptr->lchild!=NULL)

{

ptrsave=ptr;

ptr=ptr->lchild;

}

suc=ptr;

parsuc=ptrsave;

if(suc->lchild==NULL && suc->rchild==NULL)

case\_a(parsuc,suc);

else

case\_b(parsuc,suc);

if(par==NULL) /\*if item to be deleted is root node1 \*/

root=suc;

else

if(loc==par->lchild)

par->lchild=suc;

else

par->rchild=suc;

suc->lchild=loc->lchild;

suc->rchild=loc->rchild;

}/\*End of case\_c()\*/

int del(int item)

{

struct node \*parent,\*location;

if(root==NULL)

{

printf("Tree empty");

return 0;

}

find(item,&parent,&location);

if(location==NULL)

{

printf("Item not present in tree");

return 0;

}

if(location->lchild==NULL && location->rchild==NULL)

case\_a(parent,location);

if(location->lchild!=NULL && location->rchild==NULL)

case\_b(parent,location);

if(location->lchild==NULL && location->rchild!=NULL)

case\_b(parent,location);

if(location->lchild!=NULL && location->rchild!=NULL)

case\_c(parent,location);

free(location);

return 0;

}/\*End of del()\*/

int preorder(struct node \*ptr)

{

if(root==NULL)

{

printf("Tree is empty");

return 0;

}

if(ptr!=NULL)

{

printf("\n%s\t",(ptr->brand));

printf("%s",(ptr->model));

printf("\t%s\t\t",(ptr->connectivity));

printf("%s\t",(ptr->processor));

printf("\t%s",(ptr->os));

printf(" \t%d ",(ptr->ram));

printf("\t%d",(ptr->rom));

printf("\t%.1f",(ptr->fcam));

printf("\t%.1f",(ptr->bcam));

printf("\t %.1f",(ptr->disp));

printf("\t %d",(ptr->price));

preorder(ptr->lchild);

preorder(ptr->rchild);

}

return 0;

}/\*End of preorder()\*/

void inorder(struct node \*ptr)

{

if(root==NULL)

{

printf("Tree is empty");

return;

}

if(ptr!=NULL)

{

inorder(ptr->lchild);

printf("\n%s\t",(ptr->brand));

printf("%s",(ptr->model));

printf("\t%s\t\t",(ptr->connectivity));

printf("%s\t",(ptr->processor));

printf("\t%s",(ptr->os));

printf(" \t%d ",(ptr->ram));

printf("\t%d",(ptr->rom));

printf("\t%.1f",(ptr->fcam));

printf("\t%.1f",(ptr->bcam));

printf("\t %.1f",(ptr->disp));

printf("\t %d",(ptr->price))

inorder(ptr->rchild);

}

}/\*End of inorder()\*/

void postorder(struct node \*ptr)

{

if(root==NULL)

{

printf("Tree is empty");

return;

}

if(ptr!=NULL)

{

postorder(ptr->lchild);

postorder(ptr->rchild);

printf("\n%s\t",(ptr->brand));

printf("%s",(ptr->model));

printf("\t%s\t\t",(ptr->connectivity));

printf("%s\t",(ptr->processor));

printf("\t%s",(ptr->os));

printf(" \t%d ",(ptr->ram));

printf("\t%d",(ptr->rom));

printf("\t%.1f",(ptr->fcam));

printf("\t%.1f",(ptr->bcam));

printf("\t %.1f",(ptr->disp));

printf("\t %d",(ptr->price));

}

}/\*End of postorder()\*/

void display(struct node \*ptr,int level)

{

int i;

if ( ptr!=NULL )

{

display(ptr->rchild, level+1);

printf("\n\n");

for (i = 0; i < level; i++)

printf(" ");

printf("%d,%s", ptr->price,ptr->brand);

display(ptr->lchild, level+1);

}/\*End of if\*/

}/\*End of display()\*/

void search(int item)

{

struct node \*ptr;

if(root==NULL) /\*tree empty\*/

{

printf("\n MOBILE not found");

return;

}

if(item==root->price) /\*item is at root\*/

{

printf("\nbrand\tmodel\tconnection\tprocessor\tOS RAM GB ROM GB camera\tbackcamera display\tprice");

printf("\n%s\t",(root->brand));

printf("%s",(root->model));

printf("\t%s\t\t",(root->connectivity));

printf("%s\t",(root->processor));

printf("\t%s",(root->os));

printf(" \t%d ",(root->ram));

printf("\t%d",(root->rom));

printf("\t%.1f",(root->fcam));

printf("\t%.1f",(root->bcam));

printf("\t %.1f",(root->disp));

printf("\t %d",(root->price));

return;

}

/\*Initialize ptr and ptrsave\*/

if(item<root->price)

ptr=root->lchild;

else

ptr=root->rchild;

while(ptr!=NULL)

{

if(item==ptr->price)

{ printf("\nbrand\tmodel\tconnection\tprocessor\tOS RAM GB ROM GB camera\tbackcamera display\tprice");

printf("\n%s\t",(ptr->brand));

printf("%s",(ptr->model));

printf("\t%s\t\t",(ptr->connectivity));

printf("%s\t",(ptr->processor));

printf("\t%s",(ptr->os));

printf(" \t%d ",(ptr->ram));

printf("\t%d",(ptr->rom));

printf("\t%.1f",(ptr->fcam));

printf("\t%.1f",(ptr->bcam));

printf("\t %.1f",(ptr->disp));

printf("\t %d",(ptr->price));

return;

}

if(item<ptr->price)

{

ptr=ptr->lchild;

printf("\nbrand\tmodel\tconnection\tprocessor\tOS RAM GB ROM GB camera\tbackcamera display\tprice");

printf("\n%s\t",(ptr->brand));

printf("%s",(ptr->model));

printf("\t%s\t\t",(ptr->connectivity));

printf("%s\t",(ptr->processor));

printf("\t%s",(ptr->os));

printf(" \t%d ",(ptr->ram));

printf("\t%d",(ptr->rom));

printf("\t%.1f",(ptr->fcam));

printf("\t%.1f",(ptr->bcam));

printf("\t %.1f",(ptr->disp));

printf("\t %d",(ptr->price));

}

else

{ ptr=ptr->rchild;

printf("\nbrand\tmodel\tconnection\tprocessor\tOS RAM GB ROM GB camera\tbackcamera display\tprice");

printf("\n%s\t",(ptr->brand));

printf("%s",(ptr->model));

printf("\t%s\t\t",(ptr->connectivity));

printf("%s\t",(ptr->processor));

printf("\t%s",(ptr->os));

printf(" \t%d ",(ptr->ram));

printf("\t%d",(ptr->rom));

printf("\t%.1f",(ptr->fcam));

printf("\t%.1f",(ptr->bcam));

printf("\t %.1f",(ptr->disp));

printf("\t %d",(ptr->price));

}

}

}

void modify(struct node \*ptr)

{

int item,m;

printf("\n Enter the price of mobile handset to modify:");

scanf("%d",&item);

if(root==NULL) /\*tree empty\*/

{

printf("\n tree is empty");

return;

}

if(item==root->price)

{

do

{

printf("\n 1].Brand :");

printf("\n 2].model :");

printf("\n 3].connectivity :");

printf("\n 4].processor :");

printf("\n 5].operating system:");

printf("\n 6].RAM :");

printf("\n 7].ROM :");

printf("\n 8].front camera:");

printf("\n 9].Back camera:");

printf("\n 10].display:");

printf("\n 11].price :");

printf("\n 12].Exit :");

printf("\n Enter Choice That u want to Modity :");

scanf("%d",&m);

switch(m)

{

case 1:

printf("\nEnter brand Name:");

scanf("%s",(ptr->brand));

break;

case 2:

printf("\nEnter model :");

scanf("%s",(ptr->model));

break;

case 3:

printf("\nEnter connectivity type :");

scanf("%s",(ptr->connectivity));

break;

case 4:

printf("\nEnter processor :");

scanf("%s",(ptr->processor));

break;

case 5:

printf("\nEnter operating system :");

scanf("%s",(ptr->os));

break;

case 6:

printf("\nEnter RAM :");

scanf("%d",&(ptr->ram));

break;

case 7:

printf("\nEnter ROM :");

scanf("%d",&(ptr->rom));

break;

case 8:

printf("\nEnter Front camera:");

scanf("%f",&(ptr->fcam));

break;

case 9:

printf("\nEnter Back camera:");

scanf("%f",&(ptr->bcam));

break;

case 10:

printf("\nEnter size of display:");

scanf("%f",&(ptr->disp));

break;

case 11:

printf("\n Enter price of handset:");

scanf("%d",&(ptr->price));

break;

case 12: return;

}

}

while(m!=13);

}

if(item<root->price)

ptr=root->lchild;

else

ptr=root->rchild;

while(ptr!=NULL)

{

if(item==ptr->price)

{

do

{

printf("\n 1].Brand :");

printf("\n 2].model :");

printf("\n 3].connectivity :");

printf("\n 4].processor :");

printf("\n 5].operating system:");

printf("\n 6].RAM :");

printf("\n 7].ROM :");

printf("\n 8].front camera:");

printf("\n 9].Back camera:");

printf("\n 10].display:");

printf("\n 11].price :");

printf("\n 12].Exit :");

printf("\n Enter Choice That u want to Modity :");

scanf("%d",&m);

switch(m)

{

case 1:

printf("\nEnter brand Name:");

scanf("%s",(ptr->brand));

break;

case 2:

printf("\nEnter model :");

scanf("%s",(ptr->model));

break;

case 3:

printf("\nEnter connectivity type :");

scanf("%s",(ptr->connectivity));

break;

case 4:

printf("\nEnter processor :");

scanf("%s",(ptr->processor));

break;

case 5:

printf("\nEnter operating system :");

scanf("%s",(ptr->os));

break;

case 6:

printf("\nEnter RAM :");

scanf("%d",&(ptr->ram));

break;

case 7:

printf("\nEnter ROM :");

scanf("%d",&(ptr->rom));

break;

case 8:

printf("\nEnter Front camera:");

scanf("%f",&(ptr->fcam));

break;

case 9:

printf("\nEnter Back camera:");

scanf("%f",&(ptr->bcam));

break;

case 10:

printf("\nEnter size of display:");

scanf("%f",&(ptr->disp));

break;

case 11:

printf("\n Enter price of handset:");

scanf("%d",&(ptr->price));

break;

case 12:

return;

}

}

while(m!=13);

}

if(item<ptr->price)

{

do

{

printf("\n 1].Brand :");

printf("\n 2].model :");

printf("\n 3].connectivity :");

printf("\n 4].processor :");

printf("\n 5].operating system:");

printf("\n 6].RAM :");

printf("\n 7].ROM :");

printf("\n 8].front camera:");

printf("\n 9].Back camera:");

printf("\n 10].display:");

printf("\n 11].price :");

printf("\n 12].Exit :");

printf("\n Enter Choice That u want to Modity :");

scanf("%d",&m);

switch(m)

{

case 1:

printf("\nEnter brand Name:");

scanf("%s",(ptr->brand));

break;

case 2:

printf("\nEnter model :");

scanf("%s",(ptr->model));

break;

case 3:

printf("\nEnter connectivity type :");

scanf("%s",(ptr->connectivity));

break;

case 4:

printf("\nEnter processor :");

scanf("%s",(ptr->processor));

break;

case 5:

printf("\nEnter operating system :");

scanf("%s",(ptr->os));

break;

case 6:

printf("\nEnter RAM :");

scanf("%d",&(ptr->ram));

break;

case 7:

printf("\nEnter ROM :");

scanf("%d",&(ptr->rom));

break;

case 8:

printf("\nEnter Front camera:");

scanf("%f",&(ptr->fcam));

break;

case 9:

printf("\nEnter Back camera:");

scanf("%f",&(ptr->bcam));

break;

case 10:

printf("\nEnter size of display:");

scanf("%f",&(ptr->disp));

break;

case 11:

printf("\n Enter price of handset:");

scanf("%d",&(ptr->price));

break;

case 12:

return;

}

}

while(m!=13);

}

else

{

do

{

printf("\n 1].Brand :");

printf("\n 2].model :");

printf("\n 3].connectivity :");

printf("\n 4].processor :");

printf("\n 5].operating system:");

printf("\n 6].RAM :");

printf("\n 7].ROM :");

printf("\n 8].front camera:");

printf("\n 9].Back camera:");

printf("\n 10].display:");

printf("\n 11].price :");

printf("\n 12].Exit :");

printf("\n Enter Choice That u want to Modity :");

scanf("%d",&m);

switch(m)

{

case 1:

printf("\nEnter brand Name:");

scanf("%s",(ptr->brand));

break;

case 2:

printf("\nEnter model :");

scanf("%s",(ptr->model));

break;

case 3:

printf("\nEnter connectivity type :");

scanf("%s",(ptr->connectivity));

break;

case 4:

printf("\nEnter processor :");

scanf("%s",(ptr->processor));

break;

case 5:

printf("\nEnter operating system :");

scanf("%s",(ptr->os));

break;

case 6:

printf("\nEnter RAM :");

scanf("%d",&(ptr->ram));

break;

case 7:

printf("\nEnter ROM :");

scanf("%d",&(ptr->rom));

break;

case 8:

printf("\nEnter Front camera:");

scanf("%f",&(ptr->fcam));

break;

case 9:

printf("\nEnter Back camera:");

scanf("%f",&(ptr->bcam));

break;

case 10:

printf("\nEnter size of display:");

scanf("%f",&(ptr->disp));

break;

case 11:

printf("\n Enter price of handset:");

scanf("%d",&(ptr->price));

break;

case 12:

return;

}

}while(m!=13);

}

}

}

int main()

{

int choice,num;

root=NULL;

while(1)

{

printf("\t\t//////////////////////DIGITAL STORM MOBILES////////////////////////////////\n");

printf("\n");

printf("1.Insert\n");

printf("2.Delete\n");

printf("3.Inorder Traversal\n");

printf("4.Preorder Traversal\n");

printf("5.Postorder Traversal\n");

printf("6.Display\n");

printf("7.Search\n");

printf("8.Modify\n");

printf("9.Quit\n");

printf("10.create(Queue)\n");

printf("11.display(Queue)\n12.delete(Queue)\n13.insert(Queue)\n");

printf("Enter your choice : ");

scanf("%d",&choice);

//system("cls");

switch(choice)

{

case 1:

printf("\n Enter the price of mobile handset to be insert:");

scanf("%d",&num);

insert(num);

break;

case 2:

printf("\n Enter the price of mobile handset to be deleted:");

scanf("%d",&num);

del(num);

break;

case 3:

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

printf("\nbrand\tmodel\tconnection\tprocessor\tOS RAM GB ROM GB camera\tbackcamera display\tprice");

inorder(root);

break;

case 4:

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

printf("\nbrand\tmodel\tconnection\tprocessor\tOS RAM GB ROM GB camera\tbackcamera display\tprice");

preorder(root);

break;

case 5:

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

printf("\nbrand\tmodel\tconnection\tprocessor\tOS RAM GB ROM GB camera\tbackcamera display\tprice");

postorder(root);

break;

case 6:

display(root,1);

break;

case 7:

printf("\n Enter the price of mobile handset to be searched:");

scanf("%d",&num);

search(num);

break;

case 8:

modify(root);

break;

case 9:

exit(0);

break;

case 10:

printf("Enter no. of item");

scanf("%d",&n);

create\_queue(n);

break;

case 11:

display\_queue(front);

break;

case 12:

del\_queue();

break;

case 13:

front= insert\_loc(front,rear);

rear=rear->next;

n=n+1;

break;

}/\*End of switch \*/

}/\*End of while \*/

}/\*End of main()\*/

void create\_queue()

{

node1 \*temp;

int i;

for(i=0;i<n;i++)

{

if (rear==NULL)

{

rear=(struct node1\*)malloc(sizeof(node1)\*100);

rear->next=NULL;

printf("\n Enter brand of mobile:");

scanf("%s",(rear->brand));

printf("\n Enter model name:");

scanf("%s",(rear->model));

printf("\n Enter connectivity type:");

scanf("%s",(rear->connectivity));

printf("\n Enter processor:");

scanf("%s",(rear->processor));

printf("\n Enter operating system:");

scanf("%s",(rear->os));

printf("\n Enter size of RAM in GB");

scanf("%d",&(rear->ram));

printf("\n Enter size of ROM:");

scanf("%d",&(rear->rom));

printf("\n Enter front camera:");

scanf("%f",&(rear->fcam));

printf("\n Enter back camera:");

scanf("%f",&(rear->bcam));

printf("\n Enter size of display:");

scanf("%f",&(rear->disp));

printf("\n Enter the price of mobile handset:");

scanf("%d",&(rear->price));

front=rear;

}

else

{

temp=(struct node1\*)malloc(sizeof(node1));

printf("\n Enter brand of mobile:");

scanf("%s",(temp->brand));

printf("\n Enter model name:");

scanf("%s",(temp->model));

printf("\n Enter connectivity type:");

scanf("%s",(temp->connectivity));

printf("\n Enter processor:");

scanf("%s",(temp->processor));

printf("\n Enter operating system:");

scanf("%s",(temp->os));

printf("\n Enter size of RAM in GB:");

scanf("%d",&(temp->ram));

printf("\n Enter size of ROM:");

scanf("%d",&(rear->rom));

printf("\n Enter front camera:");

scanf("%f",&(temp->fcam));

printf("\n Enter back camera:");

scanf("%f",&(temp->bcam));

printf("\n Enter size of display:");

scanf("%f",&(temp->disp));

printf("\n Enter the price of mobile handset:");

scanf("%d",&(temp->price));

rear->next=temp;

temp->next=NULL;

rear=temp;

}

}

}

void display\_queue(node1 \*front)

{

node1 \*p;

if(front==NULL)

{

printf("\n\*Queue Empty\*\n");

}

else

{

p=front;

printf("\nbrand\tmodel\tconnection\tprocessor\tOS RAM GB ROM GB camera\tbackcamera display\tprice\n");

while(p!=NULL)

{

printf("\n%s\t",p->brand);

printf("%s",p->model);

printf("\t%s\t\t",p->connectivity);

printf("%s\t",p->processor);

printf("\t%s",p->os);

printf(" \t%d ",p->ram);

printf("\t%d",p->rom);

printf("\t%.1f",p->fcam);

printf("\t%.1f",p->bcam);

printf("\t %.1f",p->disp);

printf("\t %d",p->price);

p=p->next;

}

}

}

void del\_queue()

{

node1 \*temp;

if(front==NULL)

{

printf("\n\*Queue Empty\*\n");

}

else

{

temp = front;

front=front->next;

free(temp);

}

}

node1\* insert\_loc(node1 \*front,node1 \*rear)

{

node1 \*p;

printf("Enter the new position-");

p=(node1\*)malloc(sizeof(node1));

printf("\n Enter brand of mobile:");

scanf("%s",p->brand);

printf("\n Enter model name:");

scanf("%s",p->model);

printf("\n Enter connectivity type:");

scanf("%s",p->connectivity);

printf("\n Enter processor:");

scanf("%s",p->processor);

printf("\n Enter operating system:");

scanf("%s",p->os);

printf("\n Enter size of RAM in GB:");

scanf("%d",&p->ram);

printf("\n Enter size of ROM");

scanf("%d",&(rear->rom));

printf("\n Enter front camera:");

scanf("%f",&p->fcam);

printf("\n Enter back camera:");

scanf("%f",&p->bcam);

printf("\n Enter size of display:");

scanf("%f",&p->disp);

printf("\n Enter the price of mobile handset:");

scanf("%d",&p->price);

p->next=NULL;

p->next=rear->next;

rear->next=p;

return(front);

}

**OUTPUT:**

1.Insert

2.Delete

3.Inorder Traversal

4.Preorder Traversal

5.Postorder Traversal

6.Display

7.Search

8.Modify

9.Quit

10.create(Queue)

11.display(Queue)

12.delete(Queue)

13.insert(Queue)

Enter your choice : 6

**100,apple**

**200,nokia**

1.Insert

2.Delete

3.Inorder Traversal

4.Preorder Traversal

5.Postorder Traversal

6.Display

7.Search

8.Modify

9.Quit

10.create(Queue)

11.display(Queue)

12.delete(Queue)

13.insert(Queue)

Enter your choice : 3

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

brand model connection processor OS RAM GB ROM GB camera backcamera display price

apple xr 4G bionic ios 2 256 22.0 22.0 6.2 100

nokia asha 4G meditek andriod 6 256 32.0 48.0 7.1 200

Enter your choice : 4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

brand model connection processor OS RAM GB ROM GB camera backcamera display price

apple xr 4G bionic ios 2 256 22.0 22.0 6.2 100

nokia asha 4G meditek andriod 6 256 32.0 48.0 7.1 200

//////////////////////DIGITAL STORM MOBILES////////////////////////////////

1.Insert

2.Delete

3.Inorder Traversal

4.Preorder Traversal

5.Postorder Traversal

6.Display

7.Search

8.Modify

9.Quit

10.create(Queue)

11.display(Queue)

12.delete(Queue)

13.insert(Queue)

Enter your choice : 5

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

brand model connection processor OS RAM GB ROM GB camera backcamera display price

nokia asha 4G meditek andriod 6 256 32.0 48.0 7.1 200

apple xr 4G bionic ios 2 256 22.0 22.0 6.2 100

//////////////////////DIGITAL STORM MOBILES////////////////////////////////

1.Insert

2.Delete

3.Inorder Traversal

4.Preorder Traversal

5.Postorder Traversal

6.Display

7.Search

8.Modify

9.Quit

10.create(Queue)

11.display(Queue)

12.delete(Queue)

13.insert(Queue)

Enter your choice : 7

Enter the price of mobile handset to be searched:200

brand model connection processor OS RAM GB ROM GB camera backcamera display price

nokia asha 4G meditek andriod 6 256 32.0 48.0 7.1 200

//////////////////////DIGITAL STORM MOBILES////////////////////////////////

1.Insert

2.Delete

3.Inorder Traversal

4.Preorder Traversal

5.Postorder Traversal

6.Display

7.Search

8.Modify

9.Quit

10.create(Queue)

11.display(Queue)

12.delete(Queue)

13.insert(Queue)

Enter your choice : 2

Enter the price of mobile handset to be deleted:100

//////////////////////DIGITAL STORM MOBILES////////////////////////////////

1.Insert

2.Delete

3.Inorder Traversal

4.Preorder Traversal

5.Postorder Traversal

6.Display

7.Search

8.Modify

9.Quit

10.create(Queue)

11.display(Queue)

12.delete(Queue)

13.insert(Queue)

Enter your choice : 6

**200,nokia**

//////////////////////DIGITAL STORM MOBILES////////////////////////////////

1.Insert

2.Delete

3.Inorder Traversal

4.Preorder Traversal

5.Postorder Traversal

6.Display

7.Search

8.Modify

9.Quit

10.create(Queue)

11.display(Queue)

12.delete(Queue)

13.insert(Queue)

Enter your choice : 8

Enter the price of mobile handset to modify:200

1].Brand :

2].model :

3].connectivity :

4].processor :

5].operating system:

6].RAM :

7].ROM :

8].front camera:

9].Back camera:

10].display:

11].price :

12].Exit :

Enter Choice That u want to Modity :11

Enter price of handset:100

1].Brand :

2].model :

3].connectivity :

4].processor :

5].operating system:

6].RAM :

7].ROM :

8].front camera:

9].Back camera:

10].display:

11].price :

12].Exit :

Enter Choice That u want to Modity :12

//////////////////////DIGITAL STORM MOBILES////////////////////////////////

1.Insert

2.Delete

3.Inorder Traversal

4.Preorder Traversal

5.Postorder Traversal

6.Display

7.Search

8.Modify

9.Quit

10.create(Queue)

11.display(Queue)

12.delete(Queue)

13.insert(Queue)

Enter your choice : 6

100,nokia