Binary search tree

#include<stdio.h>

#include<stdlib.h>

#include<malloc.h>

struct node

{

int data;

struct node\*left,\*right;

}\*root=NULL,\*newnode;

struct node\* create(struct node\*root,int val)

{

if(root==NULL)

{

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

newnode->data=val;

newnode->left=NULL;

newnode->right=NULL;

return(newnode);

}

else if(val<root->data)

root->left=create(root->left,val);

else if(val>root->data)

root->right=create(root->right,val);

return(root);

}

void inorder(struct node \*root)

{

if(root!=NULL)

{

inorder(root->left);

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

inorder(root->right);

}

}

void preorder(struct node \*root)

{

if(root!=NULL)

{

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

preorder(root->left);

preorder(root->right);

}

}

void postorder(struct node \*root)

{

if(root!=NULL)

{

postorder(root->left);

postorder(root->right);

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

}

}

int main()

{

int ch,val;

do{

printf("\nMain Menu");

printf("\n1.create\n2.inorder\n3.preorder\n4.postorder\n5.exit");

printf("\nenter the choice:");

scanf("%d",&ch);

switch(ch)

{

case 1:

printf("enter the element:");

scanf("%d",&val);

root=create(root,val);

break;

case 2:

inorder(root);

break;

case 3:

preorder(root);

break;

case 4:

postorder(root);

break;

case 5:

exit(0);

}

}while(ch>=1 &&ch<=5);

}

Output

Main Menu

1.create

2.inorder

3.preorder

4.postorder

5.exit

enter the choice:1

enter the element:35

Main Menu

1.create

2.inorder

3.preorder

4.postorder

5.exit

enter the choice:1

enter the element:30

Main Menu

1.create

2.inorder

3.preorder

4.postorder

5.exit

enter the choice:1

enter the element:25

Main Menu

1.create

2.inorder

3.preorder

4.postorder

5.exit

enter the choice:1

enter the element:50

Main Menu

1.create

2.inorder

3.preorder

4.postorder

5.exit

enter the choice:1

enter the element:45

Main Menu

1.create

2.inorder

3.preorder

4.postorder

5.exit

enter the choice:1

enter the element:37

Main Menu

1.create

2.inorder

3.preorder

4.postorder

5.exit

enter the choice:2

25 30 35 37 45 50

Main Menu

1.create

2.inorder

3.preorder

4.postorder

5.exit

enter the choice:3

35 30 25 50 45 37

Main Menu

1.create

2.inorder

3.preorder

4.postorder

5.exit

enter the choice:4

25 30 37 45 50 35

Main Menu

1.create

2.inorder

3.preorder

4.postorder

5.exit

enter the choice:5

--------------------------------

Process exited after 37.67 seconds

HASHING LINEAR PROBING

#include<stdio.h>

#include<stdlib.h>

#define TABLE\_SIZE 10

int h[TABLE\_SIZE]={NULL};

void insert()

{

int key,index,i,flag=0,hkey;

printf("\nenter a value to insert into hash table\n");

scanf("%d",&key);

hkey=key%TABLE\_SIZE;

for(i=0;i<TABLE\_SIZE;i++)

{

index=(hkey+i)%TABLE\_SIZE;

if(h[index] == NULL)

{

h[index]=key;

break;

}

}

if(i == TABLE\_SIZE)

printf("\nelement cannot be inserted\n");

}

void display()

{

int i;

printf("\nelements in the hash table are \n");

for(i=0;i< TABLE\_SIZE; i++)

printf("\nat index %d \t value = %d",i,h[i]);

}

main()

{

int opt,i;

while(1)

{

printf("\nPress 1. Insert\t 2. Display \t3.Exit \n");

scanf("%d",&opt);

switch(opt)

{

case 1:

insert();

break;

case 2:

display();

break;

case 3:exit(0);

}

}

}

OUTPUT

Press 1. Insert 2. Display 3.Exit

1

enter a value to insert into hash table

35

Press 1. Insert 2. Display 3.Exit

2

elements in the hash table are

at index 0 value = 0

at index 1 value = 0

at index 2 value = 0

at index 3 value = 0

at index 4 value = 0

at index 5 value = 35

at index 6 value = 0

at index 7 value = 0

at index 8 value = 0

at index 9 value = 0

Press 1. Insert 2. Display 3.Exit

3

--------------------------------

Process exited after 32.44 seconds with return value 0

Press any key to continue . . .

QUEUE

#include <stdio.h>

#include<stdlib.h>

#define MAX 5

void insert();

void delete();

void display();

int queue\_array[MAX];

int rear = - 1;

int front = - 1;

int main()

{

int choice;

while (1)

{

printf("1.Insert element to queue\ n");

printf("2.Delete element from queue \n");

printf("3.Display all elements of queue\ n");

printf("4.Quit \n");

printf("Enter your choice : ");

scanf("%d", &choice);

switch(choice)

{

case 1:

insert();

break;

case 2:

delete();

break;

case 3:

display();

break;

case 4:

exit(1);

default:

printf("Wrong choice n");

}

}

}

void insert()

{

int item;

if(rear == MAX - 1)

printf("Queue Overflow n");

else

{

if(front== - 1)

front = 0;

printf("Inset the element in queue : ");

scanf("%d", &item);

rear = rear + 1;

queue\_array[rear] = item;

}

}

void delete()

{

if(front == - 1 || front > rear)

{

printf("Queue Underflow n");

return;

}

else

{

printf("Element deleted from queue is : %dn", queue\_array[front]);

front = front + 1;

}

}

void display()

{

int i;

if(front == - 1)

printf("Queue is empty n");

else

{

printf("Queue is : n");

for(i = front; i <= rear; i++)

printf("%d ", queue\_array[i]);

printf("n");

}

}

OUTPUT

1.Insert element to queue n2.Delete element from queue

3.Display all elements of queue n4.Quit

Enter your choice : 1

Inset the element in queue : 34

1.Insert element to queue n2.Delete element from queue

3.Display all elements of queue n4.Quit

Enter your choice : 1

Inset the element in queue : 90

1.Insert element to queue n2.Delete element from queue

3.Display all elements of queue n4.Quit

Enter your choice : 1

Inset the element in queue : 67

1.Insert element to queue n2.Delete element from queue

3.Display all elements of queue n4.Quit

Enter your choice : 1

Inset the element in queue : 78

1.Insert element to queue n2.Delete element from queue

3.Display all elements of queue n4.Quit

Enter your choice : 2

Element deleted from queue is : 34n1.Insert element to queue n2.Delete element from queue

3.Display all elements of queue n4.Quit

Enter your choice : 3

Queue is : n90 67 78 n1.Insert element to queue n2.Delete element from queue

3.Display all elements of queue n4.Quit

Enter your choice : 4

--------------------------------

STACK INFIX TO POSTFIX

#include<stdio.h>

#include<ctype.h>

#include<string.h>

char stk[20];

int top=-1;

void push(char c)

{

stk[++top]=c;

}

char pop()

{

return(stk[top--]);

}

int priority(char c)

{

if(c=='^'|| c=='&' || c=='|')

return 3;

else if (c=='/'|| c=='\*' || c=='%')

return 2;

else if(c=='+' || c=='-')

return 1;

else

return 0;

}

main()

{

char in[50],post[50],ch;

int i,j,l;

printf("Enter the string :");

gets(in);

l=strlen(in);

j=0;

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

{

if(isalpha(in[i]))

post[j++]=in[i];

else

{

if(in[i]=='(')

push(in[i]);

else if(in[i]==')')

while((ch=pop())!='(')

post[j++]=ch;

else

{

while(priority(in[i])<=priority(stk[top]))

post[j++]=pop();

push(in[i]);

}

}

}

while(top!=-1)

post[j++]=pop();

post[j]='\0';

printf("\n equivalent infix to postfix is:%s",post);

}

[21:09, 30/09/2022] Jerry🥰: #include<stdio.h>

#include<stdlib.h>

#define max 3

int top=-1,stack[max];

void push();

void pop();

void peek();

void display();

int main()

{

int ch;

while(1)

{

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

printf("\n1.Push (or) 2.Pop (or) 3.peek (or) 4.display (or) 5.exit");

printf("\n\nenter your choice(1 to 5):");

scanf("%d",&ch);

switch(ch)

{

case 1: push();

break;

case 2: pop();

break;

case 3: peek();

break;

case 4: display();

break;

case 5: exit(0);

default: printf("\ninvalid");

}

}

}

void push()

{

int val;

if(top==max-1)

printf("\nstack is-full");

else

printf("add element=");

scanf("%d",&val);

top=top+1;

stack[top]=val;

}

void pop()

{

if(top==-1)

printf("\nstack is-empty");

else

printf("removed element=%d",stack[top]);

top=top-1;

}

void peek()

{

if(top==-1)

printf("stack is-empty");

else

printf("top value =%d",stack[top]);

}

void display()

{

int i;

if(top==-1)

printf("\nstack is empty");

else

printf("stack elements=\n");

for(i=top;i>=0;--i)

printf("%d\n",stack[i]);

}

OUTPUT

Stack Menu \*

1.Push (or) 2.Pop (or) 3.peek (or) 4.display (or) 5.exit

enter your choice(1 to 5):1

add element=12

\* Stack Menu \*

1.Push (or) 2.Pop (or) 3.peek (or) 4.display (or) 5.exit

enter your choice(1 to 5):1

add element=13

\* Stack Menu \*

1.Push (or) 2.Pop (or) 3.peek (or) 4.display (or) 5.exit

enter your choice(1 to 5):4

stack elements=

13

12

\* Stack Menu \*

1.Push (or) 2.Pop (or) 3.peek (or) 4.display (or) 5.exit

enter your choice(1 to 5):\*