Date: 07.11.2023 Name: Niranjan.B
Ex. No: 8 Reg. No.: 3122225001082

UCS 2312 Data Structures Lab

Assignment 8: Binary Heap And Its Applications

Date of Assignment: 07.11.2023

priorityQueueADT consists of integer element. Implement the following methods

- void insert(struct priorityQueueADT *P, int x) – Insertion new item into priority queue using Max Heap property
- int delete(struct priorityQueueADT *P) – Will remove the root of binary heap
- void display(struct priorityQueueADT *P) – Will display the contents pf Priority Queue

1. Demonstrate ADT with the following testcase

```
insert(p,14);
insert(p,16);
insert(p,22);
insert(p,11);
insert(p,9);
insert(p,18);
insert(p,10);
insert(p,7);
insert(p,4);
insert(p,1);
```

2. Write an application to design a priority queue using max binary heap. An item in the priority queue consists of employee id and salary amount. The queue supports two operations, namely, insertion and deletion.
Test the application with the following

```
insert(p,('A',15000));
insert(p,('K',12000));
insert(p,('R',4000));
insert(p,('T',3500));
insert(p,('L',4600));
insert(p,('P',6000));
insert(p,('Y',8600));
```

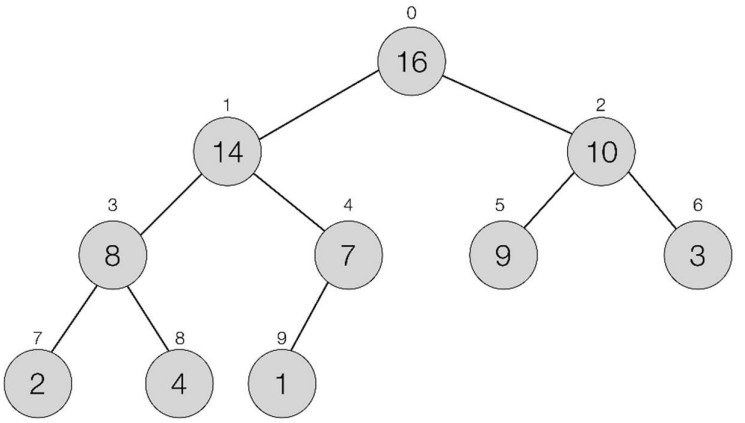
Output:

Employees are removed in the following order

('A',15000), ('K',12000), ('Y',8600), ('P',6000), ('L',4600), ('R',4000), ('T',3500),

Date: 07.11.2023 Name: Niranjan.B
Ex. No: 8 Reg. No.: 3122225001082

Data Structure – Binary Heap:



0	1	2	3	4	5	6	7	8	9
16	14	10	8	7	9	3	2	4	1

```
struct PQ
{
    int data[100];
    int size;
};
```

Algorithm –

Algorithm: Insertion new item into priority queue using Max Heap property

Input – Pointer to Priority Queue, data to be added to Priority Queue

Output – void

1. i=++P->size
2. while (P->data[i/2]<x)
   P->data[i]=P->data[i/2]
   i/=2
3. P->data[i]=x

Algorithm: Will remove the root of binary heap

Input – Pointer to Priority Queue

Output – int

1. If(P->size<1)
   return 930
2. max=P->data[1]
3. last=P-data[P->size--]
4. i=1
5. while((i*2)<=P->size))
   child=i*2

```
        if(P->data[child+1]>P->data[child])
                child++
        if(last<P->data[child])
                P->data[i]=P->data[child]
        else
                break
        i=child
```

6. P->data[i]=last
7. Return max

Algorithm: Will display the contents pf Priority Queue

Input – Pointer to Priority Queue
Output – void

1. i=1
2. while(i<=P->size)
   print P->data[i]
   i++

MaxHeap.h code:

```
#include<stdio.h>
#include<stdlib.h>
#include"AVLtree.h"

void main()
{

      struct tree* t=NULL;
      int choice=100;
      int el;
      while(choice!=4)
      {
            printf("\n\n1.Insert\n2.Print\n3.Find Parent\n4.Exit\nChoice =
");
            scanf("%d",&choice);
            switch(choice)struct PQ
{
      int data[100];
      int size;
};

void create(struct PQ* P)
{
      P->size=0;
      P->data[0]=930;
}

void insert(struct PQ* P,int x)
{
      int i;
      for(i=++P->size;P->data[i/2]<x;i/=2)
      {
            P->data[i]=P->data[i/2];
      }
```

```
	P->data[i]=x;
}

int deleteMin(struct PQ* P)
{
	if(P->size<1)
	{
		printf("Queue Empty");
		return 930;
	}
	int i,child;
	int max=P->data[1];
	int last=P->data[P->size--];
	for(i=1;(i*2)<=P->size;i=child)
	{
		child=i*2;
		if(P->data[child+1]>P->data[child])
		{
			child++;
		}
		if(last<P->data[child])
		{
			P->data[i]=P->data[child];
		}
		else
			break;
	}
	P->data[i]=last;
	return max;
}

void print(struct PQ* P)
{
	for (int i=1;i<=P->size;i++)
	printf ("(%d) ",P->data[i]);
}
		{
			case 1:
			printf("Enter the element: ");
			scanf("%d",&el);
			t=insert(t,el);
			break;
			case 2:
			hierarchical(t,0);
			printf("\n\n");
			break;
			case 3:
			printf("Enter the element: ");
			scanf("%d",&el);
			struct tree *parent=findParent(t,el);
			if(parent!=NULL)
				printf("Parent = %d",parent->data);
			else
				printf("Element Not Found");
```

```
                break;
                case 4:
                exit(0);
                break;
          }
     }
}
```

MaxHeap.c code:

```
#include<stdio.h>
#include<stdlib.h>
#include"MaxHeap.h"

void main()
{

     struct PQ* P=(struct PQ*)malloc(sizeof(struct PQ));
     int choice=1;
     int el;
     create(P);
     while(choice)
     {
          printf("\n\n1.Insert\n2.Delete
Maximum\n3.Display\n4.Exit\nChoice : ");
          scanf("%d",&choice);
          switch(choice)
          {
               case 1:
               printf("Enter the element: ");
               scanf("%d",&el);
               insert(P,el);
               break;
               case 2:
               el=deleteMin(P);
               if(el!=930)
               printf("Max Element = %d",el);
               break;
           case 3:
           printf("Elements : ");
           print(P);
               case 4:
               exit(0);
               break;
               default:
               printf("Invalid Choice");
          }
     }
}
```

Output Screen:

```
PS D:\College\Sem 3\Data Structures\Binary Heap> gcc MaxHeap.c
PS D:\College\Sem 3\Data Structures\Binary Heap> ./a.exe


1.Insert
2.Delete Maximum
3.Display
4.Exit
Choice : 1
Enter the element: 14


1.Insert
2.Delete Maximum
3.Display
4.Exit
Choice : 1
Enter the element: 16


1.Insert
2.Delete Maximum
3.Display
4.Exit
Choice : 1
Enter the element: 22


1.Insert
2.Delete Maximum
3.Display
4.Exit
Choice : 1
Enter the element: 11


1.Insert
2.Delete Maximum
3.Display
4.Exit
Choice : 1
Enter the element: 9
```

Name: Niranjan.B
Reg. No.: 3122225001082

```
1.Insert
2.Delete Maximum
3.Display
4.Exit
Choice : 1
Enter the element: 18


1.Insert
2.Delete Maximum
3.Display
4.Exit
Choice : 1
Enter the element: 10


1.Insert
2.Delete Maximum
3.Display
4.Exit
Choice : 1
Enter the element: 7


1.Insert
2.Delete Maximum
3.Display
4.Exit
Choice : 1
Enter the element: 4


1.Insert
2.Delete Maximum
3.Display
4.Exit
Choice : 1
Enter the element: 1
```

```
1.Insert
2.Delete Maximum
3.Display
4.Exit
Choice : 3
Elements : (22) (14) (18) (11) (9) (16) (10) (7) (4) (1)
```

Write an application to design a priority queue using max binary heap. An item in the priority queue consists of employee id and salary amount. The queue supports two operations, namely, insertion and deletion.

PriorityQueue.h code:

```
#include <stdio.h>

struct emp
{
      char id;
      int salary;
};

struct pq
{
      struct emp el[100];
      int size;
};

void init(struct pq*p)
{
      p->size=0;
      p->el[0].id = 'Z';
      p->el[0].salary = 10000000;
}

void print (struct pq*p)
{
      for (int i=1;i<=p->size;i++)
      printf ("(%c,%d)\t", p->el[i].id, p->el[i].salary);
}

void insert (struct pq*p, char id, int sal)
{
      int i;
      for (i=++p->size;p->el[i/2].salary<sal;i/=2)
      {
            p->el[i].id = p->el[i/2].id;
            p->el[i].salary = p->el[i/2].salary;
      }
      p->el[i].id = id;
      p->el[i].salary = sal;
}

struct emp deletemax (struct pq*p)
{
      int i,child;
      struct emp maxelt = p->el[1];
      struct emp lastelt = p->el[p->size--];
      for (i=1;(i*2)<=p->size;i=child)
      {
            child = i*2;
            if (p->el[child+1].salary>p->el[child].salary)
            child++;
```

```
            if (lastelt.salary < p->el[child].salary)
            p->el[i] = p->el[child];
            else
            break;
      }
      p->el[i] = lastelt;
      return maxelt;
}
```

PriorityQueue.c code:

```
#include <stdio.h>
#include <stdlib.h>
#include "priorityQ.h"

void main ()
{
      struct pq*p = (struct pq*)malloc(sizeof(struct pq));
      init (p);
      int d, ch=-1; char x;
      struct emp e;
      while (ch!=0)
      {
            printf ("\nMENU:\n1 - Insert\n2 - Delete maximum element\n3 -
Print\n0 - Exit\nEnter your choice: ");
            scanf ("%d", &ch);
            scanf ("%c", &x);
            switch (ch)
            {
                  case 1:
                  {
                        printf ("Enter element to insert: ");
                        scanf ("%c", &e.id);
                        scanf ("%c", &x);
                        scanf ("%d", &e.salary);
                        insert (p,e.id,e.salary);
                        break;
                  }
                  case 2:
                  {
                        e = deletemax (p);
                        printf ("\n(%c,%d) is deleted.\n", e.id,e.salary);
                        break;
                  }
                  case 3:
                  {
                        print (p);
                        break;
                  }
                  case 0:
                  {
                        printf ("Quitting ...\n");
                        break;
                  }
                  default:
                  printf ("\nInvalid choice. Enter again.\n");
```

```
                }
        }
}
```

Output Screen:

```
PS D:\College\Sem 3\Data Structures\Binary Heap> gcc priorityQ.c
PS D:\College\Sem 3\Data Structures\Binary Heap> ./a.exe

MENU:
1 - Insert
2 - Delete maximum element
3 - Print
0 - Exit
Enter your choice: 1
Enter element to insert: A
15000

MENU:
1 - Insert
2 - Delete maximum element
3 - Print
0 - Exit
Enter your choice: 1
Enter element to insert: K
12000

MENU:
1 - Insert
2 - Delete maximum element
3 - Print
0 - Exit
Enter your choice: 1
Enter element to insert: R
4000

MENU:
1 - Insert
2 - Delete maximum element
3 - Print
0 - Exit
Enter your choice: 1
Enter element to insert: T
3500
```

Name: Niranjan.B
Reg. No.: 3122225001082

```
MENU:
1 - Insert
2 - Delete maximum element
3 - Print
0 - Exit
Enter your choice: 1
Enter element to insert: L
4600

MENU:
1 - Insert
2 - Delete maximum element
3 - Print
0 - Exit
Enter your choice: 1
Enter element to insert: P
6000

MENU:
1 - Insert
2 - Delete maximum element
3 - Print
0 - Exit
Enter your choice: 1
Enter element to insert: Y
8600

MENU:
1 - Insert
2 - Delete maximum element
3 - Print
0 - Exit
Enter your choice: 2

(A,15000) is deleted.
```

```
MENU:
1 - Insert
2 - Delete maximum element
3 - Print
0 - Exit
Enter your choice: 2

(K,12000) is deleted.

MENU:
1 - Insert
2 - Delete maximum element
3 - Print
0 - Exit
Enter your choice: 2

(Y,8600) is deleted.

MENU:
1 - Insert
2 - Delete maximum element
3 - Print
0 - Exit
Enter your choice: 2

(P,6000) is deleted.

MENU:
1 - Insert
2 - Delete maximum element
3 - Print
0 - Exit
Enter your choice: 2

(L,4600) is deleted.
```

Name: Niranjan.B
Reg. No.: 3122225001082

```
MENU:
1 - Insert
2 - Delete maximum element
3 - Print
0 - Exit
Enter your choice: 2

(R,4000) is deleted.

MENU:
1 - Insert
2 - Delete maximum element
3 - Print
0 - Exit
Enter your choice: 2

(T,3500) is deleted.
```

Learning Outcome:

Learning Outcome		
Design	3	Design of Binary Heap is clear
Understanding of DS	3	Understood Binary Max Heap and
Use of DS	3	Binary Min Heap applications
Debugging	3	Was able to fix errors
Best Practices		
Design before coding	3	Designed properly
Use of algorithmic notation	2	Can be improved
Use of multifile C program	3	Used multiple files
Versioning of code	2	Can be versioned properly