In-Course Examination

SY. BTECH
SEM - III

SUB: DATA AND FILE STRUCTURE

Cycle-II

PAPER SOLUTION

Q-1 [a] (I) The worst case occurs when the elements of INFO are in increasing order, where each comparison of MAx with INFO[K] forces ~~INFO~~ POS and MAx to be updated. In this case,

$F(n) = n-1$ [1]

Example : INFO = {81, 85, 98, 99} [1]

N = 4

POS and MAx will be updated for 3 times $n-1$.

(II) The best case occurs when the largest element appears first and so when the comparison of MAx with INFO[K] never forces POS and MAx to be updated. Accordingly in this case

$F(n) = 0$ [1]

Example : INFO = {155, 81, 91, 101} [1]

N = 4

with MAx = INFO [1]

POS and MAx will never be updated.

Q-1(III) There are six possible ways the elements can appear in INFO, which correspond to $3! = 6$ permutation of 1,2,3. e.g. For each permutation P, let n_p denote the number of times LOOP and MAX are updated when the algorithm is executed with input P. The six permutations P and the corresponding values n_p follow:

Permutation P:	123	132	213	231	312	321
Value of n_p:	2	1	1	1	0	0

[2]

Assuming all permutations P are equally likely

$$F(n) = \frac{2+1+1+1+0+0}{6} = \frac{5}{6}$$

$$F(n) = \frac{5}{6}$$

[1]

Q.1 [b] STEP TABLE- [2]

	Statement	S/e	Frequency	Total Steps
1	sum (X, Y, Z, m, n)	0	-	0
2	{	0	-	0
3	for (i=1 to m) do	1	m+1	(m+1)
4	for (j=1 to n) do	1	m(n+1)	m(n+1)
5	Z[i,j] = X[i,j] + Y[i,j]	1	m*n	m*n
6	}	0	-	0
			Total	(m+1) + m(n+1) + m*n = 2mn + 2m + 1

[1]

Q-2

```
/*
INPUT: Array YEAR
OUTPUT:(a). Years in which no customer was born.
       (b). Number of years in which no customer was born.
       (c). Number of customers who are atleast 50 year old at
the end of 2017.
*/
#include<iostream.h>
#include<conio.h>
int YEAR[]={10,4,7,0,7,8,12,5,0,1,6,8,0,5,5,8,9,12,0,0,3,5,11};

void NOCUST()
{
for(int i=0;i<20;i++) {
if(YEAR[i]==0)
{
cout<<"\nYear="<<i+1950;
}
}
}

void COUNTNOCUST()
{
int count=0;
for(int i=0;i<20;i++) {
if(YEAR[i]==0)
{
count++;
}
}
cout<<"Count="<<count;
}

void OLD50()
{
int count=0;
for(int i=0;i<18;i++)
{
count=count+YEAR[i];
}
cout<<"Total no. of customer who are at least 50 at the end of
2017:"<<count;
}

void main()
{
clrscr();
cout<<"\nYear with no customer born"<<endl;
NOCUST();
cout<<"\nNo. of years with no customer born"<<endl;
COUNTNOCUST();
cout<<"\nNo. of 50 year old customer"<<endl;
OLD50();
getch();
}
```

[3] for three Algorithm

[2]

[2]

[3]

Algorithm Insert() [Q.3]

INPUT - Name & DFS_marks of student [1]

OUTPUT - Sorted list after Insertion

Datastructure - sorted linked list

~~Steps;~~

```
class node
{   char name[20];
    int marks;
};
class list
{ public:
      node *head;
      void insert();
      void del();
};
```

[1] [2]

Steps of Insertion [3]

```
1.- node *curr, *prev, *temp;
2   prev = NULL; curr = head;
3   temp = new node; temp->next = NULL
4   cout << "\n Enter name to be added";
5   cin >> temp->name;
6   cout << " Enta marks ";
7   cin >> temp->marks;
8   while (curr != NULL)
    {
        if (curr->marks > temp->marks)
        {
            if (prev == NULL)
            {   temp->next = head;
```

```
else
    {
        prev->next = temp;
        temp->next = curr;
    }
} // end of first If
else
    {
        prev = curr;
        curr = curr->next;
    }
}
```

Steps for deletion [3]

```
1  node *curr, *prev;
2  int key;  cin>>key;
3  curr = head;  prev = NULL;
4  while (curr != NULL) {
     if (curr->marks == key)
     {
        if (prev == NULL)
        {  head = curr->next;
           delete curr; break;
        }
        else
        {  prev->next = curr->next;
           delelete curr;
           break;
        }
     }
     else
        {  prev = curr;
           curr = curr->next;
        }
}
```

Q-4

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
#include<string.h>
//  Creating a NODE Structure
class node
{  public:
   char data;
   struct node *next;
};

// Creating a class STACK
class stack
{
   struct node *top;
   public:
   stack() // constructure
   {
         top=NULL;
   }
   void push(char); // to insert an element
   char pop();  // to delete an element
   void show(); // to show the stack
};
// PUSH Operation
void stack::push(char value)
{
   struct node *ptr;
   ptr=new node;
   ptr->data=value;
   ptr->next=NULL;
   if(top!=NULL)
         ptr->next=top;
   top=ptr;
}

// POP Operation
char stack::pop()
{
   struct node *temp;
   if(top==NULL)
   {
         cout<<"\nThe stack is empty!!!";
   }
   temp=top;
   top=top->next;
   cout<<"\nPOP Operation.....nPoped value is
"<<temp->data;
   return(temp->data);
}

// Show stack
void stack::show()
{
   struct node *ptr1=top;
   cout<<"\nThe stack is\n";
   while(ptr1!=NULL)
   {
         cout<<ptr1->data<<" ->";
         ptr1=ptr1->next;
   }
   cout<<"NULL\n";
}

// Main function
int main()
{
   stack s;
   char str[10];int len,count=0;
   cin>>str;
   int choice;
   len = strlen(str);
   cout<<len;

   for (int i = 0; i < len; i++)
   {
     s.push(str[i]);
   }

   for (i = 0; i < len; i++)
   {
         if (str[i] == s.pop())
           count++;
   }
   cout<<count;
   if (count == len)
         printf("%s is a Palindrome string\n", str);
   else
         printf("%s is not a palindrome string\n", str);
 getch();
 return 0;
}
```

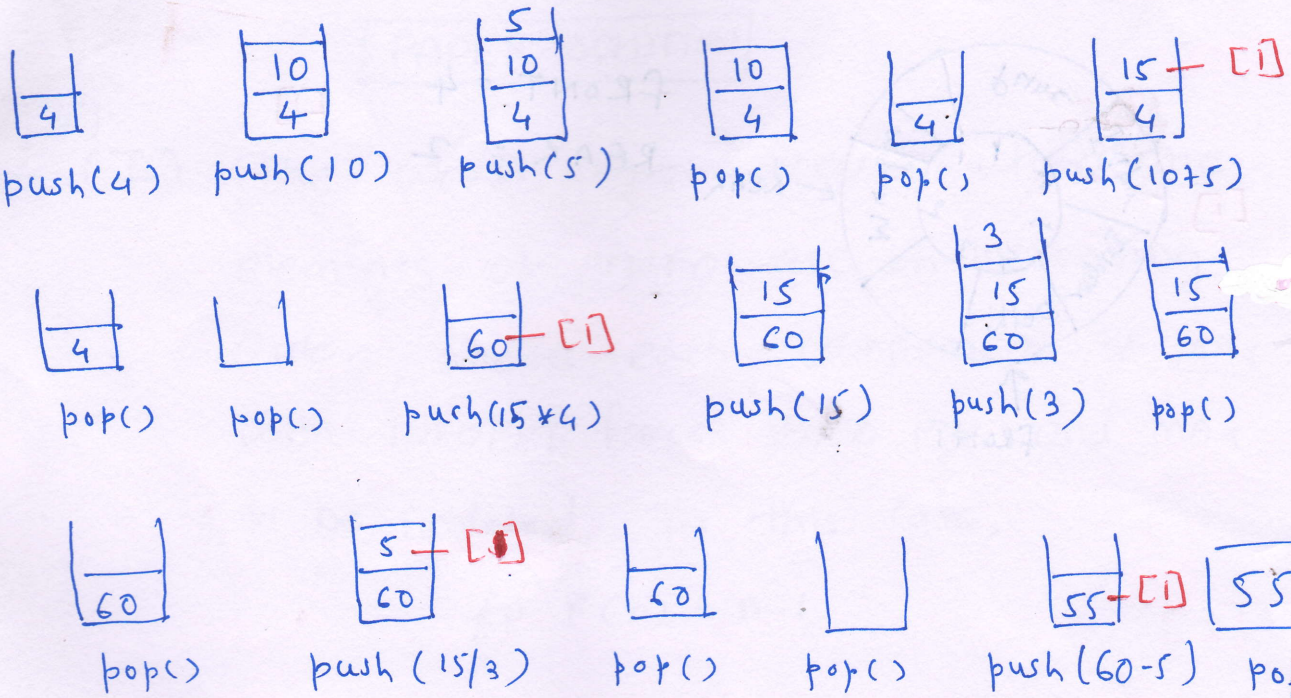
[1]

[3]

[3]

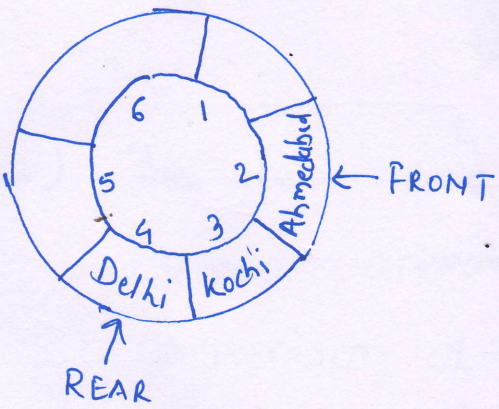
[3]

[a] Expression 4, 10, 5, +, *, 15, 3, /, −



value of expression 55 [1]

[b] Given circular queue

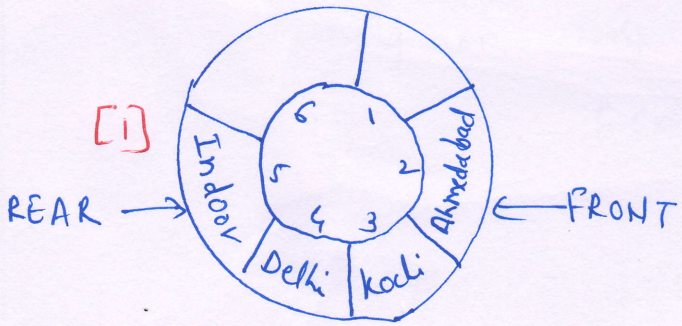


FRONT = 2

REAR = 4

1) Indore is added

[1]



FRONT = 2

REAR = 5

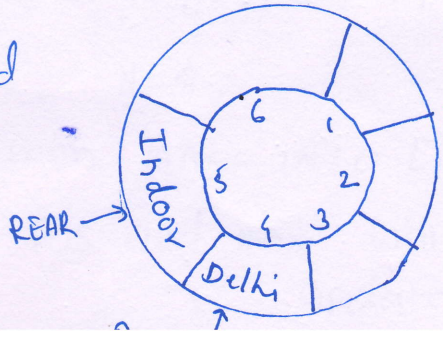
[0.5]

2) Two cities deleted

[1]

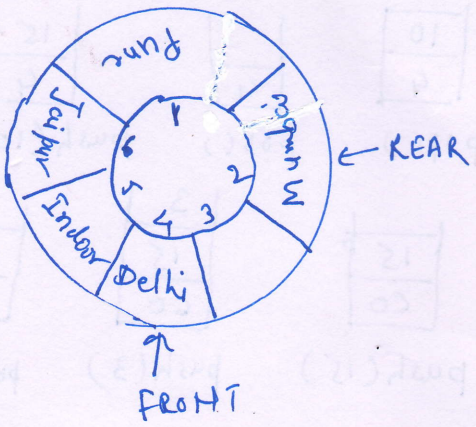
FRONT = 4

REAR = 5



[0.5]

5) Jaipur, Pune, Mumbai added



[1]

FRONT = 4
REAR = 2

[1]