

**SIDDHI SINGH**

**17BIT0028**

**ITE 1004**

**DSA EXERCISE – 3**

When burning a DVD it is essential that the laser beam burning pits onto the surface is constantly fed with data, otherwise the DVD fails. Most leading DVD burn applications make use of a circular buffer to stream data from the hard disk onto the DVD. The first part, the 'writing process' fills up a circular buffer with data, then the 'burning process' begins to read from the buffer as the laser beam burns pits onto the surface of the DVD. If the buffer starts to become empty, the application should continue filling up the emptied space in the buffer with new data from the disk. Implement this scenario using Circular Queue.

CODE

```
#include<stdio.h>
#define max 5
int front = -1, rear = -1;

char buff[max];

char DVD[100];

int n = 0;

int
isempty ()
{
    if (front == -1)
        return 1;
    else
        return 0;
}

int
isfull ()
{
    if (front == (rear + 1) % max)
```

```
return 1;
```

```
    else
```

```
return 0;
```

```
}
```

```
int
```

```
enqueue (char x)
```

```
{
```

```
    if (isfull ())
```

```
    {
```

```
printf ("BUFFER IS FULL\n");
```

```
return 0;
```

```
}
```

```
    if (front == -1)
```

```
    {
```

```
rear = 0;
```

```
front = 0;
```

```
}
```

```
    else
```

```
    {
```

```
rear = (rear + 1) % max;
```

```
}
```

```
buff[rear] = x;
```

```
return 1;
```

```
}
```

```
char
```

```
dequeue ()
```

```
{
```

```
char x;
```

```
    if (isempty ())
```

```
{
printf ("BUFFER IS EMPTY\n");
return 0;
}

x = buff[front];
if (front == rear)
{
front = -1;
rear = -1;
}
else
{
front = (front + 1) % max;
}
return x;
}

void
display ()
{

int i = front;
if (isempty ())
{
printf ("BUFFER IS EMPTY\n");
}
else
{
while (i != rear)
{
```

```
printf ("%c\n", buff[i]);
```

```
i = (i + 1) % max;
```

```
}
```

```
printf ("%c\n", buff[i]);
```

```
}
```

```
}
```

```
void  
display1 ()  
{
```

```
int k = 0;
```

```
for (k = 0; k < n; k++)
```

```
printf ("%c\n", DVD[k]);
```

```
}
```

```
int  
main ()  
{
```

```
char s[100], a, c;
```

```
int ch = 0, j = 0, y = 0, k = 0;
```

```
printf ("ENTER THE EXPRESSION\n");
```

```
gets (s);
```

```
while (ch != 5)
```

```
{
```

```
printf ("1 READ THE CHAR AND STORE IT IN BUFFER\n");
```

```
printf ("2 WRITE INTO DVD\n");
```

```
printf ("3 DISPLAY THE CURRENT CONTENTS OF BUFFER\n");
```

```
printf ("4 DISPLAY THE CURRENT CONTENTS OF DVD\n");
```

```
printf ("5 EXIT\n");
```

```
scanf ("%d", &ch);
```

```
switch (ch)
```

```
{  
case 1:  
a = s[j];  
y = enqueue (a);  
if (y == 1)  
j++;  
break;  
case 2:  
c = dequeue ();  
DVD[n++] = c;  
a = s[j];  
y = enqueue (a);  
if (y == 1)  
j++;  
break;  
case 3:  
display ();  
break;  
case 4:  
display1 ();  
break;  
case 5:  
break;  
}  
}  
}
```

```

1  #include<stdio.h>
2  #define max 5
3  int front = -1, rear = -1;
4
5  char buff[max];
6
7  char DVD[100];
8
9  int n = 0;
10
11 int
12 isempty ()
13 {
14
15 if (front == -1)
16
17 return 1;
18
19 else
20
21 return 0;
22
23 }
24
25
26 int
27 isfull ()
28 {
29
30 if (front == (rear + 1) % max)
31
32 return 1;
33
34 else
35
36 return 0;
37
38 }

```

```

37
38 }
39
40
41 int
42 enqueue (char x)
43 {
44
45 if (isfull ())
46
47 {
48
49 printf ("BUFFER IS FULL\n");
50
51 return 0;
52
53 }
54
55 if (front == -1)
56
57 {
58
59 rear = 0;
60
61 front = 0;
62
63 }
64
65 else
66
67 {
68
69 rear = (rear + 1) % max;
70
71 }
72
73 buff[rear] = x;
74
75 return 1;

```

```

74
75     return 1;
76
77 }
78
79
80
81 char
82 dequeue ()
83 {
84
85     char x;
86
87     if (isempty ())
88     {
89
90
91         printf ("BUFFER IS EMPTY\n");
92
93         return 0;
94     }
95
96
97     x = buff[front];
98
99     if (front == rear)
100
101     {
102
103         front = -1;
104
105         rear = -1;
106
107     }
108
109     else
110
111     {

```

```

110
111     {
112
113         front = (front + 1) % max;
114
115     }
116
117     return x;
118
119 }
120
121
122 void
123 display ()
124 {
125
126
127     int i = front;
128
129     if (isempty ())
130
131     {
132
133         printf ("BUFFER IS EMPTY\n");
134
135     }
136
137     else
138
139     {
140
141         while (i != rear)
142
143         {
144
145             printf ("%c\n", buff[i]);
146
147             i = (i + 1) % max;

```

```

148 }
149 |
150 |
151 printf ("%c\n", buff[i]);
152 |
153 }
154 |
155 }
156 |
157 |
158 void
159 display1 ()
160 {
161 |
162 |
163 int k = 0;
164 |
165 for (k = 0; k < n; k++)
166 |
167 printf ("%c\n", DVD[k]);
168 |
169 }
170 |
171 |
172 int
173 main ()
174 {
175 |
176 char s[100], a, c;
177 |
178 int ch = 0, j = 0, y = 0, k = 0;
179 |
180 printf ("ENTER THE EXPRESSION\n");
181 |
182 gets (s);
183 |
184 while (ch != 5)
185 |

```

```

185 |
186 {
187 |
188 printf ("1 READ THE CHAR AND STORE IT IN BUFFER\n");
189 |
190 printf ("2 WRITE INTO DVD\n");
191 |
192 printf ("3 DISPLAY THE CURRENT CONTENTS OF BUFFER\n");
193 |
194 printf ("4 DISPLAY THE CURRENT CONTENTS OF DVD\n");
195 |
196 printf ("5 EXIT\n");
197 |
198 scanf ("%d", &ch);
199 |
200 switch (ch)
201 |
202 {
203 |
204 case 1:
205 |
206 a = s[j];
207 |
208 y = enqueue (a);
209 |
210 if (y == 1)
211 |
212 j++;
213 |
214 break;
215 |
216 case 2:
217 |
218 c = dequeue ();
219 |
220 DVD[n++] = c;
221 |
222 a = s[j];

```



```

216 case 2:
217
218 c = dequeue ();
219
220 DVD[n++] = c;
221
222 a = s[j];
223
224 y = enqueue (a);
225
226 if (y == 1)
227
228 j++;
229
230 break;
231
232 case 3:
233
234 display ();
235
236 break;
237
238 case 4:
239
240 display1 ();
241
242 break;
243
244 case 5:
245
246 break;
247
248 }
249
250 }
251
252 }
253

```

## OUTPUT

```

ENTER THE EXPRESSION
Siddhi
1 READ THE CHAR AND STORE IT IN BUFFER
2 WRITE INTO DVD
3 DISPLAY THE CURRENT CONTENTS OF BUFFER
4 DISPLAY THE CURRENT CONTENTS OF DVD
5 EXIT
1
1 READ THE CHAR AND STORE IT IN BUFFER
2 WRITE INTO DVD
3 DISPLAY THE CURRENT CONTENTS OF BUFFER
4 DISPLAY THE CURRENT CONTENTS OF DVD
5 EXIT
2
1 READ THE CHAR AND STORE IT IN BUFFER
2 WRITE INTO DVD
3 DISPLAY THE CURRENT CONTENTS OF BUFFER
4 DISPLAY THE CURRENT CONTENTS OF DVD
5 EXIT
3
1
1 READ THE CHAR AND STORE IT IN BUFFER
2 WRITE INTO DVD
3 DISPLAY THE CURRENT CONTENTS OF BUFFER
4 DISPLAY THE CURRENT CONTENTS OF DVD
5 EXIT
4
S

```

```
5 EXIT
1
1 READ THE CHAR AND STORE IT IN BUFFER
2 WRITE INTO DVD
3 DISPLAY THE CURRENT CONTENTS OF BUFFER
4 DISPLAY THE CURRENT CONTENTS OF DVD
5 EXIT
2
1 READ THE CHAR AND STORE IT IN BUFFER
2 WRITE INTO DVD
3 DISPLAY THE CURRENT CONTENTS OF BUFFER
4 DISPLAY THE CURRENT CONTENTS OF DVD
5 EXIT
3
1
1 READ THE CHAR AND STORE IT IN BUFFER
2 WRITE INTO DVD
3 DISPLAY THE CURRENT CONTENTS OF BUFFER
4 DISPLAY THE CURRENT CONTENTS OF DVD
5 EXIT
4
5
1 READ THE CHAR AND STORE IT IN BUFFER
2 WRITE INTO DVD
3 DISPLAY THE CURRENT CONTENTS OF BUFFER
4 DISPLAY THE CURRENT CONTENTS OF DVD
5 EXIT
5
```