DATA STRUCTURE USING ARRAY

Multiple Choice Type Questions

1. The best data structure to see whether an arithmetic expression has balanced parenthesis is a [WBUT 2006, 2007, 2011, 2017]
   a) queue b) stack c) tree d) list

Answer: (b)

2. What is the minimum number of stacks of size *n* required to implement a queue of size *n*? [WBUT 2006]
   a) One b) Two c) three d) Four

Answer: (b)

3. A data structure in which an element is added and removed only from one end is known as [WBUT 2006, 2007]
   a) queue b) stack
   c) in-built data structure d) none of these

Answer: (b)

4. FIFO concept is applicable in [WBUT 2007]
   a) Stack b) Queue c) Linked list d) All of These

Answer: (b)

5. The sparse matrix have [WBUT 2007, 2009]
   a) Many zero entries b) many non- zero entries
   c) Higher dimension d) none of these

Answer: (a)

6. The expression which access the (i, j) th entry of a m × n matrix stored in column major form is [WBUT 2007, 2008]
   a) $n*(i-1)+j$ b) $m*(j-1)+i$ c) $m*(n-j)+j$ d) $n*(m-i)+j$

Answer: (b)

7. The operation for adding an entry to a stack is traditionally called [WBUT 2007]
   a) add b) append c) insert d) push

Answer: (d)

8. An adjacency matrix representation of a graph cannot contain information of [WBUT 2007, 2015, 2016, 2018]

a) Nodes
b) edges
c) direction of edges
d) parallel edges

Answer: (d)

9. The time required to insert an element in a stack with linked list implementation is [WBUT 2008]

a) 0 (1)
b) $0 \log_2 n$
c) 0 (n)
d) none of these

Answer: (c)

10. The postfix expression for the infix operation $\frac{A+B*(C+D)}{F+D*E}$ is [WBUT 2008]

a) $\frac{AB+CD+*F}{D+E*}$
b) $\frac{ABCD+*F}{+DE*+}$
c) $\frac{A*B+CD}{F*DE++}$
d) None of these

Answer: (d)

11. A linear list in which elements can be added on removed at either end but not in the middle is known as [WBUT 2008]

a) queue
b) deque
c) stack
d) tree

Answer: (b)

12. Which of the following types of expressions does not require precedence rules for evaluation? [WBUT 2008]

a) fully parenthesized infix expression
b) partially parenthesized infix expression
c) both (a) and (b)
d) prefix expression

Answer: (a)

13. When an element is inserted in queue, the position of front [WBUT 2009]

a) increments
b) decrements
c) unchanged
d) none of these

Answer: (c)

14. The following sequence of operations is performed on a stack push(1), push(2), pop, push(1), push(2), pop, pop, pop, push(2), pop. The sequence of popped out values is [WBUT 2010, 2015]

a) 2, 2, 1, 2, 1
b) 2, 2, 1, 1, 2
c) 2, 1, 2, 2, 1
d) 2, 1, 2, 2, 2

Answer: (b)

15. To make a queue empty, elements can be deleted, till [WBUT 2010]
a) front = rear + 1 b) front = rear –1 c) front = rear d) none of these

Answer: (c)

16. What is the time complexity if insert an element into stack implemented by linked list? [WBUT 2010]
a) $O(n)$ b) $O(1)$ c) $O(n^2)$ d) none of these

Answer: (b)

17. Let q be the queue of integers defined as follows [WBUT 2011, 2015, 2016]

```
#define max10
Struct queue
{ int data [MAX];
      int rear, front;
} q;
```

To insert an element into the queue, we may write operation
a) ++q.data[q.rear]=x b) q.data[q.rear]++=x
c) q.data[++q.rear]=x d) none of these

Answer: (c)

18. Stack works on [WBUT 2011]
a) LIFO b) FIFO c) both (a) & (b) d) none of these

Answer: (a)

19. The memory address of the first element of an array is called [WBUT 2012]
a) floor address b) foundation address
c) first address d) base address

Answer: (d)

20. The memory address of fifth element of an array can be calculated by the formula [WBUT 2012]
a) LOC (Array [5] =Base (Array) + w(5-lower bound), where w is the number of words per memory cell for the array
b) LOC (Array [5] =Base (Array [5]) + (5-lower bound), where w is the number of words per memory cell for the array
c) LOC (Array [5]) = Base (Array [4]) + (5-Upper bound), where w is the number of words per memory cell for the array
d) none of these

Answer: (a)

21. Which of the following data structure can't store the non-homogeneous data elements? [WBUT 2012]

a) Arrays b) Records c) Pointers d) None of these

Answer: (a)

22. Which of the following is a collection of items into which items can be inserted arbitrarily and from which only the smallest item can be removed? [WBUT 2013, 2014]

a) Descending priority queue b) Ascending priority

c) Fifo queue d) none of these

Answer: (b)

23. Which of the following data structures may give overflow error, even though the current number of elements in it is less than its size? [WBUT 2013, 2014, 2015]

a) Simple queue b) Circular queue

c) Priority queue d) none of these

Answer: (a)

24. Adjacency matrix for an undirected graph is [WBUT 2015]

a) unit matrix b) symmetric matrix

c) asymmetric matrix d) none of these

Answer: (b)

25. Which of the following data structures may give overflow error, even though the current number of elements in it, is less than its size? [WBUT 2016]

a) Simple queue b) Circular queue

c) Stack d) None of these

Answer: (a)

26. The following sequence of operations is performed on a stack: [WBUT 2016]
push(1), push(2), pop, push(1), push(2), pop, pop, pop, push(2), pop.
The sequence of popped values is

a) 2, 2, 1, 2, 1 b) 2, 2, 1, 1, 2 c) 2, 1, 2, 2, 1 d) 2, 1, 2, 2, 2

Answer: (b)

27. A data structure where elements can be added or removed at either end but not in the middle is [WBUT 2017]

a) linked list b) stack c) queue d) dequeue

Answer: (d)

28. Which of the following data structures may give over flow error, even though the current number of elements in it is less than its size? [WBUT 2018]
a) Simple queue b) Circular queue
c) Priority queue d) None of these
Answer: (a)

Short Answer Type Questions

1. Define stack data structure. [WBUT 2004]
OR,
What is Stack? [WBUT 2010, 2015]
Answer:
Refer to Question No. 4 of Long Answer Type Questions.

2. Write C-like algorithms / C-functions to implement 'push' and 'pop' operations in stack. [WBUT 2004, 2013]
OR,
Write the algorithm of push ( ) and pop ( ) operation in STACK. [WBUT 2007, 2010, 2012]
OR,
Write push and pop function for a stack. [WBUT 2014]
OR,
Explain various operations performed using stack with examples. [WBUT 2015]
OR,
Write a C function to implement 'PUSH' and 'POP' operations in a stack. [WBUT 2016, 2017]

Answer:
Push and pop algorithms

push()
1. if top of the stack is greater than equal to maximum number of entries into the stack, then print "Stack is already full. Cannot add more items."
2. top of stack = t
3. increment the top of the stack

pop()
1. decrement top of the stack (top := top -1)
2. if top <0,then print "Stack is already empty. No items to read."

3. What are the steps required to check parenthesis? [WBUT 2006]

Answer:

Stack can be used to check parenthesis. The steps are as follows.

- ⇨ For each left parenthesis, do a *push*
- ⇨ For each right parenthesis, do a *pop* and make sure the item popped is equal to the right parenthesis (same type)
- ⇨ If the stack is empty when doing a pop → ERROR
- ⇨ If the stack is not empty when the expression ends → ERROR

4. Convert the following infix expression to corresponding postfix expression:
4 + 3 * 10/6 + 7 – 4/2 + 5 ^ 3 [WBUT 2007, 2013, 2016]

Answer:

Symbol scanned	Stack	Output
4	NULL	4
+	+	4
3	+	4 3
*	+*	4 3
10	+*	4 3 10
/	+/	4 3 10 *
6	+/	4 3 10 * 6
+	+	4 3 10 * 6 / +
7	+	4 3 10 * 6 / + 7
-	+	4 3 10 * 6 / + 7 -
4	+	4 3 10 * 6 / + 7 – 4
/	+/	4 3 10 * 6 / + 7 – 4
2	+/	4 3 10 * 6 / + 7 - 4 2
+	+	4 3 10 * 6 / + 7 - 4 2 / +
5	+	4 3 10 * 6 / + 7- 4 2 / + 5
^	+^	4 3 10 * 6 / + 7- 4 2 / + 5
3		4 3 10 * 6 / + 7 - 4 2 / + 5 3 ^ +
NONE	NONE	4 3 10 * 6 / + 7 – 4 2 / + 5 3 ^ +

Postfix Expression: 4 3 10 * 6 / + 7 – 4 2 / + 5 3 ^ +

5. What is a sparse matrix? Give an example.

Answer: [WBUT 2008]

A **sparse matrix** is a matrix populated primarily with zeros. When storing and manipulating sparse matrices on a computer, it is beneficial and often necessary to use specialized algorithms and data structures that take advantage of the sparse structure of

the matrix. Operations using standard dense matrix structures and algorithms are slow and consume large amounts of memory when applied to large sparse matrices.
A bitmap image having only 2 colors, with one of them dominant (say a file that stores a handwritten signature) can be encoded as a sparse matrix that contains only row and column numbers for pixels with the non-dominant color.

6. Convert the following Infix to postfix expression using Stack: [WBUT 2008, 2012]
(A + (B * C – (D / E – F) * G) * H)

Answer:

Symbol scanned	Stack	Output
(	(	-
A	(	A
+	(+	A
(	(+(	A
B	(+(	AB
*	(+(*	AB
C	(+(*	ABC
-	(+(-	ABC*
(	(+(-(	ABC*
D	(+(-(	ABC*D
/	(+(-(/	ABC*D
E	(+(-(/	ABC*DE
-	(+(-(-	ABC*DE/
F	(+(-(-	ABC*DE/F
)	(+(-	ABC*DE/F-
*	(+(-*	ABC*DE/F-
G	(+(-*	ABC*DE/F-G
)	(+	ABC*DE/F-G*-
*	(+*	ABC*DE/F-G*-
H	(+*	ABC*DE/F-G*-H
)	-	ABC*DE/F-G*-H*+
NONE	NONE	ABC*DE/F-G*-H*+

7. Convert the following into postfix: **[WBUT 2009, 2018]**
a + b × c $ d – (e – f × g) / h

Answer:
a b c d $ × + e f g × - h / -

8. What is double ended queue? What are the advantages of circular queue over simple queue? [WBUT 2010]

OR,

What are the differences between normal queue and circular queue? [WBUT 2012]

Answer:

1st Part: *Refer to Question No. 8(b) of Long Answer Type Questions.*

2nd Part: *Refer to Question No. 1(a) of Long Answer Type Questions.*

9. Convert the following infix expression into postfix form by using Stack:

$a+b*c-(d-e*f)/g$ [WBUT 2011, 2017]

Answer:

Symbol scanned	Stack	Output
a	NULL	a
+	+	a
b	+	a b
*	+*	a b
c	+*	a b c
-	-	a b c*+
(	-(	a b c*+
d	-(	a b c*+d
-	-(-	a b c*+d
e	-(-	a b c*+d e
*	-(-*	a b c*+d e
f	-(-*	a b c*+d e f
)	-	a b c*+d e f*-
/	-/	a b c*+d e f*-
g	-/	a b c*+d e f*-g
NONE	NONE	a b c*+d e f*-g/-

Postfix form: abc*+def*-g/-

10. What are the overflow and underflow condition

Answer: [WBUT 2012]

A stack overflow occurs when the on--chip stack capacity is exceeded. This can be detected by a comparison of the top-pointer, the last-pointer and the index specified for a create-frame operation (for push). Initially the top-pointer is set to 1 and the last-pointer to 0, placing a dummy element on the stack.

If an index specified for a read access is greater than the number of valid on--chip stack elements, a stack underflow occurs
Queue overflow results from trying to add an element onto a full queue and queue underflow happens when trying to remove an element from an empty queue.

11. What is Dequeue? What is the advantage of Dequeue over Circular queue? **[WBUT 2012]**

Answer:

1st Part: Refer to Question No. 8(b) of Long Answer Type Questions.

2nd Part:

A circular queue is a queue in which all nodes are treated as circular such that the first node follows the last node. Both the front and the rear pointers wrap around to the beginning of the array. Items can inserted and deleted from a queue in O(1) time. Circular queue has *fixed* size. So there are no growth above limit and so there are two implementations when limit is reached:
1) discarding: push to one end of full buffer silently discards an element from other end.
2) refusing: push to full buffer fails.
On the other hand dequeues are versatile data structure than stack or queue and policy-based application (e.g. low priority go to the end, high go to the front). In a doubly linked list implementation and assuming no allocation/deallocation overhead, the time complexity of all deque operations is O(1). Additionally, the time complexity of insertion or deletion in the middle, given an iterator, is O(1); however, the time complexity of random access by index is O(n). Dequeue grows and shrinks dynamically so ends are not related to each other.

12. Write an algorithm to push an element into a queue. **[WBUT 2013]**

Answer:

Let Q[N] be an array implementation of a circular queue, of size N and type T. Then 'front' is a variable that point to the front element of the queue and 'rear' is a variable that point to the last element of the queue. Initially, rear=N-1 and front=N-1. Then the algorithm to add an element in the queue is as follows:

1. Start
2. Check the conditions

```
if(rear=N-1)THEN
  set rear to the first location,  rear=0;
else if(rear+1==front)THEN
   print "The Queue is Full";
else
  ++rear;
```

3. Add the 'item' in the Queue.

```
Q[rear]=item;
```

4. Exit

13. Convert from infix to postfix the expression:
A+(B–C/E*(F+G)–M) **[WBUT 2014]**

Answer:

Current Symbol	Operator Stack	Postfix String
A		A
+	+	A
(	+(	A
B	+(	AB
–	+(–	AB
C	+(–	ABC
/	+(–/	ABC
E	+(–/	ABCE
*	+(–/*	ABCE
(	+(–/*(	ABCE
F	+(–/*(	ABCEF
+	+(–/*(+	ABCEF
G	+(–/*(+	ABCEFG
)	+(–/*	ABCEFG+
–	+(–/–	ABCEFG+*
M	+(–/–	ABCEFG+*M
)		ABCEFG+*M–/–+

14. What is a circular queue? What are its advantages?
OR,
What is a circular queue? How it differs from linear queue? **[WBUT 2015]**

Answer: **[WBUT 2015]**

1st Part: *Refer to Question No. 3 of Long Answer Type Questions.*
2nd Part: *Refer to Question No. 1(a) of Long Answer Type Questions.*

15. What is the advantage of dequeue over ordinary queue?
Answer: **[WBUT 2017]**

Deque is advantageous in situations when there are multiple requests (customers who want their items to be billed) but the contract is such that as soon as the cashier from other counter completes all the requests he can take over the left-over requests (customers) from the opposite end of the line. This helps to use up the resource efficiently and requests are cleared at a faster rate.

Long Answer Type Questions

1. a) What is a queue? How is it different from circular queue? What advantage do we get from circular queue over ordinary queue? [WBUT 2004, 2013]

OR,

What is the problem of linear queue? How that can be solved by Circular queue? [WBUT 2004, 2011, 2013]

OR,

What advantage do we get from circular queue over ordinary queue? [WBUT 2014]

b) Write an algorithm to generate a circular queue of integer using liked structure. [WBUT 2016]

[WBUT 2004]

Answer:

a) A **Queue** is a (ordered) collection of items, where all insertions are made to the end of the sequence and all deletions always are made from the beginning of the sequence. In principle a queue is container from which data items are retrieved out in the same order they are put in.

A circular queue is a queue where the first last element of the queue points to the first element in a circular fashion.

In linear queue the condition for queue full is

QREAR==MAXLIMIT

Suppose maxlimit is ten and queue is full now if we delete 9 element from queue then inspite of queue is empty we cannot insert any element in the queue. This wastage of memory is solved through circular queue where queue full condition is

QREAR==Qfront+1

b) *Refer to Question No. 3 of Long Answer Type Questions.*

2. Give an account of row major and column major representation of two-dimensional array. [WBUT 2006]

Answer:

The memory of a computer is linear and not a matrix like a 2D array. So, the elements of the array are stored either by row, called "row-major", or by column, called "column-major". Row-major order is used most notably in C during static declaration of arrays. In C, since the length of each row is always known, the memory can be filled row one row at a time, one after the other. So the address of any specified location, for example Arr[j,k] of a two dimensional array Arr[m,n] can be calculated by using the following formula :-

(Column major order) Address(Arr[j,k])= base(Arr)+w[m(k-1)+(j–1)]

(Row major order) Address(Arr[j,k])=base(Arr)+w[n(j–1)+(k–1)]

For example, Arr(25,4) is an array with base value 200.w=4 for this array. The address of Arr(12,3) can be calculated using row-major order as

Address(Arr(12,3))=200+4[4(12–1)+(3–1)]
=200+4[4*11+2]
=200+4[44+2]
=200+4[46]
=200+184
=384

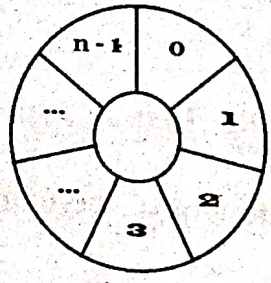
Again using column-major order

Address(Arr(12,3))= 200+4[25(3-1)+(12-1)]
=200+4[25*2+11]
=200+4[50+11]
=200+4[61]
=200+244
=444

3. What do you mean by circular queues? Give the array implementation of it. **[WBUT 2008, 2010, 2011, 2016] [WBUT 2008, 2010]**

Answer:

A circular queue can be thought of as an array of elements as a circular list where the end of the array elements [0..n-1] is glued to its front as in the following picture. n this situation, the queue indexes front and rear advance by moving them clockwise around the array.




```
/* Program of circular queue using array*/
# include
# define MAX 5
int cqueue_arr[MAX];
int front = -1;
int rear = -1;
main()  /* main function which asks users to insert, delete and
display queue */
{
int choice;
while(1)
{
```

```
printf("1.Insert\n");
printf("2.Delete\n");
printf("3.Display\n");
printf("4.Quit\n");
printf("Enter your choice : ");
scanf("%d",&choice);
switch(choice)
{
case 1 : insert(); break;
case 2 : del();break;
case 3: display(); break;
case 4: exit(1); default:
printf("Wrong choice\n");
}/*End of switch*/
}/*End of while */
}/*End of main()*/
insert()   /* insert function */
{
int added_item;
if((front == 0 && rear == MAX-1) || (front == rear+1))
{
printf("Queue Overflow \n");
return;
}
if (front == -1) /*If queue is empty */
{
front = 0;
rear = 0;
}
else
if(rear == MAX-1)/*rear is at last position of queue */
rear = 0;
else
rear = rear+1;
printf("Input the element for insertion in queue : ");
scanf("%d", &added_item);
cqueue_arr[rear] = added_item ;
}/*End of insert()*/
del()        /*delete function */
{
if (front == -1)
{
printf("Queue Underflow\n");
return ;
```

```
}
printf("Element deleted from queue is : %d\n",cqueue_arr[front]);
if(front == rear) /* queue has only one element */
{
front = -1;
rear=-1;
}
else
if(front == MAX-1)
front = 0;
else
front = front+1;
}/*End of del() */
display() / *display function */
{
int front_pos = front,rear_pos = rear;
if(front == -1)
{
printf("Queue is empty\n");
return;
}
printf("Queue elements :\n");
if( front_pos <= rear_pos )
while(front_pos <= rear_pos)
{
printf("%d ",cqueue_arr[front_pos]);
front_pos++;
}
else
{
while(front_pos <= MAX-1)
{
printf("%d ",cqueue_arr[front_pos]);
front_pos++;
}
front_pos = 0;
while(front_pos <= rear_pos)
{
printf("%d ",cqueue_arr[front_pos]);
front_pos++;
}
}/*End of else */
printf("\n");
}/*End of display() */
```

4. What are the differences between Stack and Queue? **[WBUT 2008]**

Evaluate the postfix expression:

623 + – 382/ + * 2 - 3 +

OR,

What is stack? Explain with an example. **[WBUT 2016]**

Answer:

A **Stack** is a (ordered) collection of items, where all insertions are made to the end of the sequence and all deletions always are made from the end of the sequence. In principle a stack is a container of data items, from which we get data items out in reverse order compared to the order they have been put into the container. We can also say that the item that has been put last in is coming first out. That's why a stack is also called LIFO (Last In First Out list). We can as well say that the item, which is put first in the container is get last out (First In Last Out: FILO).

A **Queue** is a (ordered) collection of items , where all insertions are made to the end of the sequence and all deletions always are made from the beginning of the sequence. In principle a queue is container from which data items are retrieved out in the same order they are put in. This means that the queue is a container that preserves the order of items put there. We can also say that the item that is put last into the queue is taken last out from the queue. We can also say that the item which is put first into the queue is taken first out. (First In First Out: FIFO).

Symbol scanned	Stack	Operation
6	6	-
2	62	-
3	623	-
+	65	2+3
-	1	6-5
3	13	-
8	138	-
2	1382	-
/	134	8/2
+	17	3+4
*	7	1*7
2	72	-
-	5	7-2
3	53	-
+	8	5+3
NONE	-	8 (Answer)

5. Write an algorithm for insertion and deletion of elements from the circular queue. **[WBUT 2010]**

Answer:

```
void CQInsert(int item)
{
  if (rear == N-1)
  {
    printf("Queue Is Full");
    return;
  }
  CQ[++rear] = item;
  if (front ==  - 1)
    front = 0;
}
int CQDelete()
{
  int x;
  if (front ==  - 1)
  {
    printf("Queue Is Empty");
    exit(0);
  }
  x = CQ[front];
  CQ[front] =  - 1;
  if (front == rear)
    front = rear =  - 1;
  else
    front++;
  return x;
}
```

6. Write an algorithm for conversion of an infix arithmetic expression in its corresponding postfix form.

Answer: **[WBUT 2015]**

Scan the Infix string from left to right.

- Initialise an empty stack.
- If the scannned character is an operand, add it to the Postfix string. If the scanned character is an operator and if the stack is empty Push the character to stack.
- If the scanned character is an Operand and the stack is not empty, compare the precedence of the character with the element on top of the stack (topStack). If topStack has higher precedence over the scanned character Pop the stack else Push the scanned character to stack. Repeat this step as long as stack is not empty and topStack has precedence over the character.

- Repeat this step till all the characters are scanned.
- (After all characters are scanned, we have to add any character that the stack may have to the Postfix string.) If stack is not empty add topStack to Postfix string and Pop the stack. Repeat this step as long as stack is not empty.

Return the Postfix string.

7. a) Convert the following infix expression to corresponding postfix expression: (A+B)/C*E+F$G–H/(I*J) [WBUT 2015]

b) Write a program to implement queue using linked list.

Answer:

a)

Current Symbol	Operator Stack	Postfix String
(	(	
A	(	A
+	(+	A
B	(+	AB
)		AB+
/	/	AB+
C	/	AB+C
*	*	AB+C/
E	*	AB+C/E
+	+	AB+C/E*
F	+	AB+C/E*F
$	+	AB+C/E*F$
G	+	AB+C/E*F$G
-	-	AB+C/E*F$G+
H	-	AB+C/E*F$G+H
/	-/	AB+C/E*F$G+H
(	+/(	AB+C/E*F$G+H
I	+/(	AB+C/E*F$G+HI
*	+/(*	AB+C/E*F$G+HI
J	+/(*	AB+C/E*F$G+HIJ
)		AB+C/E*F$G–HIJ*/-

AB+C/E*F$G–HIJ*/+ is the converted expression.

b)

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct Node
{
        int Data;
        struct Node* next;
}*rear, *front;
void delQueue()
{
        struct Node *temp, *var=rear;
       if(var==rear)
       {
              rear = rear->next;
              free(var);
       }
       else
       printf("\nQueue Empty");
}
void push(int value)
{
     struct Node *temp;
     temp=(struct Node *)malloc(sizeof(struct Node));
     temp->Data=value;
     if (front == NULL)
     {
           front=temp;
           front->next=NULL;
           rear=front;
    }
    else
    {
           front->next=temp;
           front=temp;
           front->next=NULL;
    }
}
void display()
{
     struct Node *var=rear;
     if(var!=NULL)
     {
          printf("\nElements are as:  ");
```

```
        while(var!=NULL)
        {
                printf("\t%d",var->Data);
                var=var->next;

        }
printf("\n");
}
else
printf("\nQueue is Empty");

}
int main()
{   int i=0;
front=NULL;
printf(" \n1. Push to Queue");
printf(" \n2. Pop from Queue");
printf(" \n3. Display Data of Queue");
printf(" \n4. Exit\n");
while(1)
{
    printf(" \nChoose Option: ");
    scanf("%d",&i);
    switch(i)
    {
            case 1:
            {
                int value;
                printf("\nEnter a valueber to push into
Queue: ");
                scanf("%d",&value);
                push(value);
                display();
                break;
            }
            case 2:
            {
                delQueue();
                display();
                break;
            }
            case 3:
            {
                display();
                break;
```

```
        }
        case 4:
        {
              exit(0);
        }
        default:
        {
              printf("\nwrong choice for operation");
        }}}}
```

8. Write short notes on the following:

a) Queue **[WBUT 2007]**

b) Double ended queue **[WBUT 2008, 2009]**

OR,

De-queue **[WBUT 2011, 2014, 2017]**

c) Circular queue **[WBUT 2009]**

d) Priority queue **[WBUT 2011, 2017]**

e) Sparse matrix **[WBUT 2015]**

Answer:

a) Queue:

The queue data structure is characterised by the fact that additions are made at the end, or *tail*, of the queue while removals are made from the front, or *head*, of the queue. For this reason, a queue is referred to as a FIFO structure (First-In First-Out). Queues occur naturally in situations where the rate at which clients' demand for services can exceed the rate at which these services can be supplied. In an operating system with a GUI, applications and windows communicate using messages, which are placed in message queues until they can be handled.

The main primitive operations of a queue are known as:

Add adds a new node

Remove removes a node

Additional primitives can be defined:

IsEmpty reports whether the queue is empty

IsFull reports whether the queue is full

Initialise creates/initialises the queue

Destroy deletes the contents of the queue (may be implemented by re-initialising the queue).

b) Double ended queue:

A double-ended queue is an abstract data type similar to an ordinary queue, except that it allows you to insert and delete from both sides.

It supports the following operations: *enq_front* (inserting at the front), *enq_back* (inserting at the back), *deq_front* (deleting from the front), *deq_back* (deleting from the back), and *empty* (empty queue).
By choosing a subset of these operations, one can make the double-ended queue behave like a stack or like a queue. For instance, if only *enq_front* and *deq_front* are used then we get a stack, and if only *enq_front* and *deq_back* are used then we get a queue.

c) Circular queue: ***Refer to Question No. 3 of Long Answer Type Questions.***

d) Priority queue:
A priority queue is essentially a list of items in which each item has associated with it a *priority*. In general, different items may have different priorities and we speak of one item having a higher priority than another. Given such a list we can determine which is the highest (or the lowest) priority item in the list. Items are inserted into a priority queue in any, arbitrary order. However, items are withdrawn from a priority queue in order of their priorities starting with the highest priority item first.
Two elements with the same priority are processed according to the order in which they were added to the queue.

e) Sparse matrix: ***Refer to Question No. 5 of Short Answer Type Questions.***