TOPICWISE : PROGRAMMING AND DATA STRUCTURES-1 (GATE - 2019) - REPORTS

OVERALL ANALYSIS   COMPARISON REPORT   SOLUTION REPORT

ALL(17)   CORRECT(5)   INCORRECT(5)   SKIPPED(7)

Q. 1

Analyse the code fragment given below in which size represent the size of array named as value:

```
for(int i = 0; i < size – 1; i++) {
    minindex = i;
for(int j = 0; j < size; j++) {
    if(value [j] < value [minindedx]) {
    minindex = j;
    }
}
swap(value, i, minindex);
}
```

Which of the following sorting algorithm represented by above code?

Have any Doubt ?

A Insertion sort — Your answer is **Wrong**

B Selection sort — Correct Option

Solution :
(b)
The code represent is the selection sort algorithm on an array.

C Bucket sort

D Linked list

QUESTION ANALYTICS +

Q. 2

Which of the following data structure is efficient to implement priority queue with basic operation such as insertion, deletion and searching?

FAQ Have any Doubt ?

A Linked list — Your answer is **Wrong**

B Heap — Correct Option

Solution :
(b)

Priority queue:

1. **Via Linked list:** Insertion = $O(n)$, Deletion = $O(1)$, Search = $O(n)$
2. **Via Sorted array:** Insertion = $O(n)$, Deletion = $O(1)$, Search = $O(\log n)$
3. **Via Unsorted array:** Insertion = $O(1)$, Deletion = $O(n)$, Search = $O(n)$
4. **Via Heap list:** Insertion = $O(\log n)$, Deletion = $O(\log n)$, Search = $O(\log n)$

C Sorted array

D Unsorted array

QUESTION ANALYTICS +

Q. 3

Consider the function given below, which should return the index of first zero in input array of length 'n' if present else return –1.

```
int index of zero (int[ ] array, int n) {
for (int i = 0; [P] ; i++);
    if (i = = n)
        return –1;
    return i;
}
```

Which of the should be place in code at [P] ,so that code will work fine?

FAQ Have any Doubt ?

A array[i]! = 0 && i ≤ n

B array[i]! = 0 && i < n — Your answer is **Correct**

Solution :
(b)
For every index in input array we need to check given index contain '0' or not if current index contains 0 then get out of loop and print index and if current index do not contains 0 then check it for the next index element.

array[i]! = 0

Also check index should be less than total number of elements in array i.e.

i < n

So, condition must be array[i]! = 0 && i < n.

C ! array[i] = 0 && i < n

D ! array[i] = = 0 || i < n

QUESTION ANALYTICS +

Q. 4

Consider a single array A[0..... n – 1] is used to implement two stacks. Two stacks grows from opposite ends of the array. Variables top1 and top2 points to the location of the top most element in each of the stacks with initial values of –1 and n respectively and top1 < top2 always. If certain push and pop operations are performed at either end, then which of the following represents the number of elements are present in the array at any time?

FAQ Have any Doubt ?

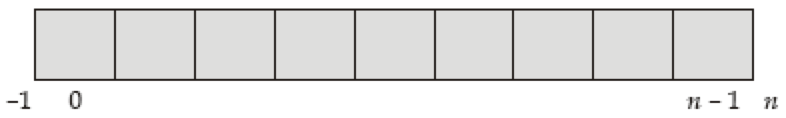
A top1 – top2 + n

B n – top2 + top1

C n + 1 – top2 + top1 Correct Option

Solution :
(c)

Consider array representation of stacks:

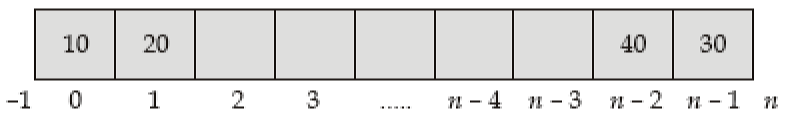


top1 = –1 represents no element in stack –1
top2 = n represents no element in stack –2

So, check option one by one when both stacks are empty:

(a) $-1 - n + n = -1$ not possible
(b) $n - n + -1 = -1$ not possible
(c) $n + 1 - n + (-1) = 0$ only possible option
(d) $n - 1 - n + (-1) = -2$ not possible

Now consider for both stack has '2' elements each:



Apply in option (c)

$$= n + 1 - (n - 2) + 1$$
$$= n + 1 - n + 2 + 1$$
$$= 4$$

So, option (c) is correct.

D n – 1 – top2 + top1

QUESTION ANALYTICS +

Q. 5

Consider the following program:

```
#include <stdio.h>
int main ( ) {
    char arr[6] = {10, 20, 30, 40, 50, 60};
    char *ptr = (char*) (& arr + 1);
    printf("%d%d", *(arr + 1), *(ptr – 1));
}
```

Which of the following represent the output of above program?

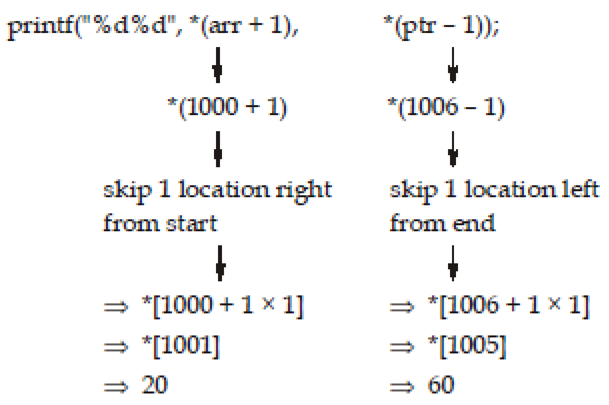
FAQ Have any Doubt ?

A 20, 60 Correct Option

Solution :
(a)

arr[6] =	10	20	30	40	50	60
	1000	1001	1002	1003	1004	1005

ptr = (char) (&arr + 1);
= [1000 + 1 × 6]
= [1006]


printf("%d%d", *(arr + 1), *(ptr – 1));

*(1000 + 1)
skip 1 location right from start
⇒ *[1000 + 1 × 1]
⇒ *[1001]
⇒ 20

*(1006 – 1)
skip 1 location left from end
⇒ *[1006 + 1 × 1]
⇒ *[1005]
⇒ 60

'arr' represents address of arr[0]
'&arr' represents the array as a whole
'&arr + 1' represents address after the last element of array
Hence '&arr + 1' points to address location 1006.

B 20, 10 Your answer is **Wrong**

C 10, 60

D Garbage value

QUESTION ANALYTICS +

Q. 6

Consider the following program:

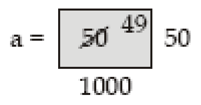
```
#include <stdio.h>
int main ( ) {
     int a = 50;
switch (a) {
     default: a = 45;
     case 49: a++;
     case 50: a−−;
     case 51: a = a + 1;
}
     printf("%d\n", a);
     return 0;
}
```

The output of above program is ________.

FAQ Have any Doubt ?

● 50 Your answer is **Correct**50

Solution :
50



case 50: a−−; a = 49
case 51: a = a + 1; a = 50
printf(a) = 50

Note:

- Since default case is above from case 50 (running case), so cannot be evaluated, but if default case after case 50 then it will be evaluated.
- After case 50, value of a is 49 but next case is 51, so it will be evaluated only.

QUESTION ANALYTICS +

Q. 7

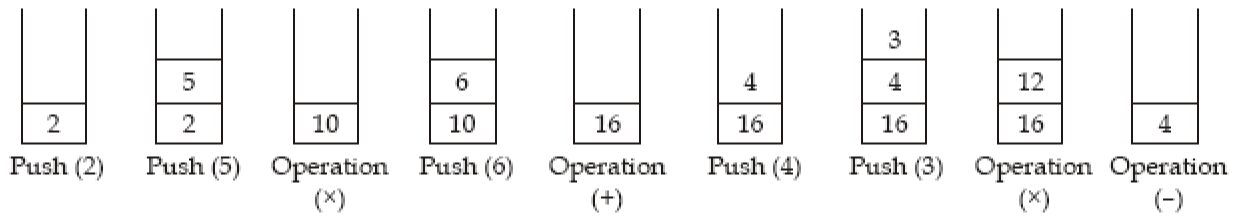
Consider a stack is used to evaluate fully parenthesized arithmetic expression from left to right. Each operand is placed on the stack and operators operate on top two elements of the stack. The minimum size of stack required to evaluate given expression is ________.
(((2 × 5) + 6) – (4 × 3))

FAQ Have any Doubt ?

● 3 Your answer is **Correct**3

Solution :
3

To evaluate an expression we need an operand stack as given in question.



So, minimum stack size needed is 3.

QUESTION ANALYTICS +

Q. 8

Consider the following C program:

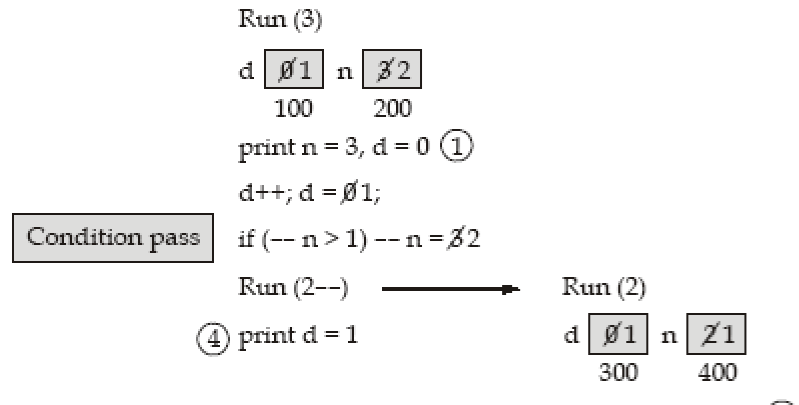
```
#include <stdio.h>
void Run (int n) {
     int d = 0;
     printf("%d", n);
     printf("%d", d++);
     if (– –n > 1)
          Run (n– –);
     printf("%d", d);
}
void main ( ) {
     Run (3);
}
```

The output of above C program is ________.

FAQ Have any Doubt ?

● 302011 Correct Option

Solution :
302011



print n = 2, d = 0 (2)

d++; d = ~~0~~1;

Condition fail if (-- n > 1) -- n = ~~2~~1

Run (1--)

(3) print d = 1

Your Answer is 31210

QUESTION ANALYTICS +

Q. 9

Consider the following recursive program:

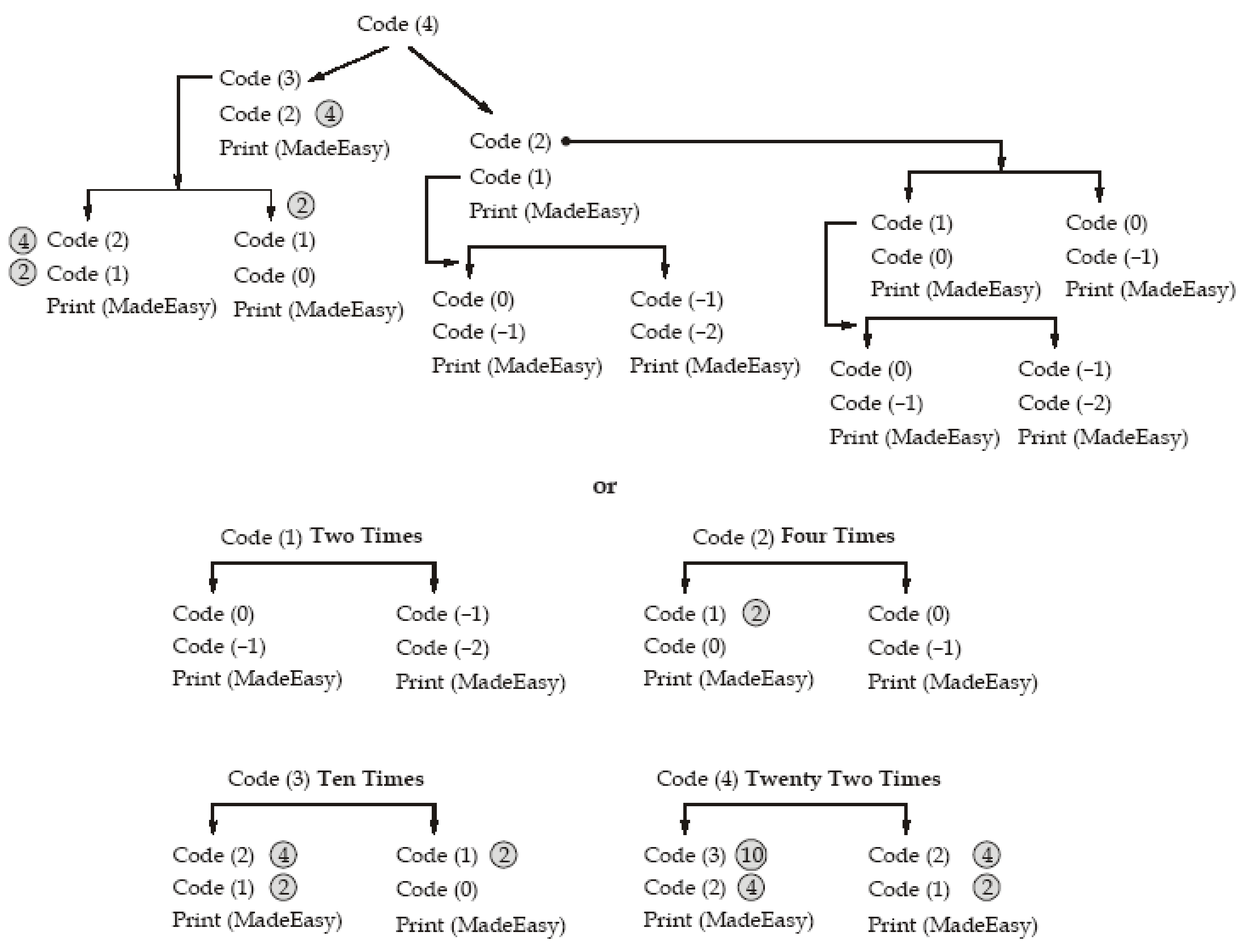
```
  #include <stdio.h>
  int main ( ) {
        code (4);
  return 0;
  }
        int code (int m) {
            if (m > 0) {
            int i = 1;
            for (; i < 3; i ++) {
                code (m – i);
                code (m – i – 1);
            printf("MadeEasy");
            }
        }
        return 0;
  }
```

The number of times "MadeEasy" will be printed ________.

FAQ Have any Doubt ?

22 Correct Option

Solution :
22



QUESTION ANALYTICS +

Q. 10

Consider the following C-program:

```
#include <stdio.h>
int main ( ) {
char *arr[ ] = {"GATE", "CAT", "IES", "IAS", "PSU", "IFS"};
      call (arr);
      return 0;
}
void call (char **ptr ) {
      char ** ptr1;
      ptr1 = (ptr+ = size of (int)) –2;
      printf("%s\n", *ptr1);
}
```

Which of the following represents the output of above program? (Assume size of int, pointer is 4B)

FAQ Have any Doubt ?

A IES Correct Option

Solution :
(a)

arr[ ] =	GATE%	CAT%	IES%	IAS%	PSU%	IFS%
	1000	1004	1008	1012	1016	1020

$$
\begin{aligned}
**ptr &= arr \Rightarrow **ptr = 1000; \\
*ptr1 &= (ptr+ = size\ of\ (int))\ [-2]; \\
&= (1000 + 4)\ [-2] \\
&= [1000 + 4 \times 4]\ [-2] \\
&= [1016]\ [-2]
\end{aligned}
$$

= [1015 – 2 × 4]
*ptr1 = [1008]
print(*ptr1) = IES

B IAS

C CAT

D PSU

QUESTION ANALYTICS +

Q. 11

Consider a stack implementation supports, in addition to PUSH and POP, an operation REVERSE, which reverses the order of the elements on the stack. Which of the following represents the minimum stack operations required to implement ENQUEUE and DEQUEUE operations of queue data structure respectively?

FAQ Have any Doubt ?

A 1 and 3

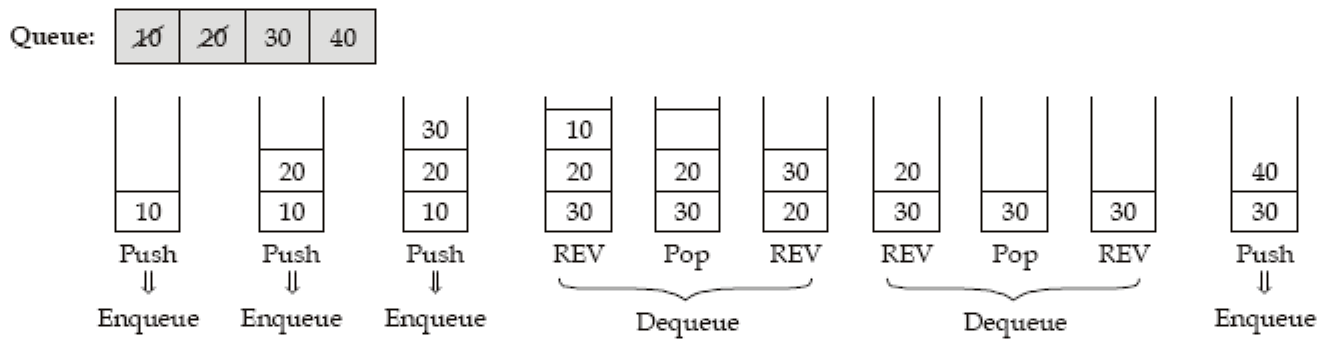
B 3 and 1 Your answer is **Wrong**

C 2 and 2

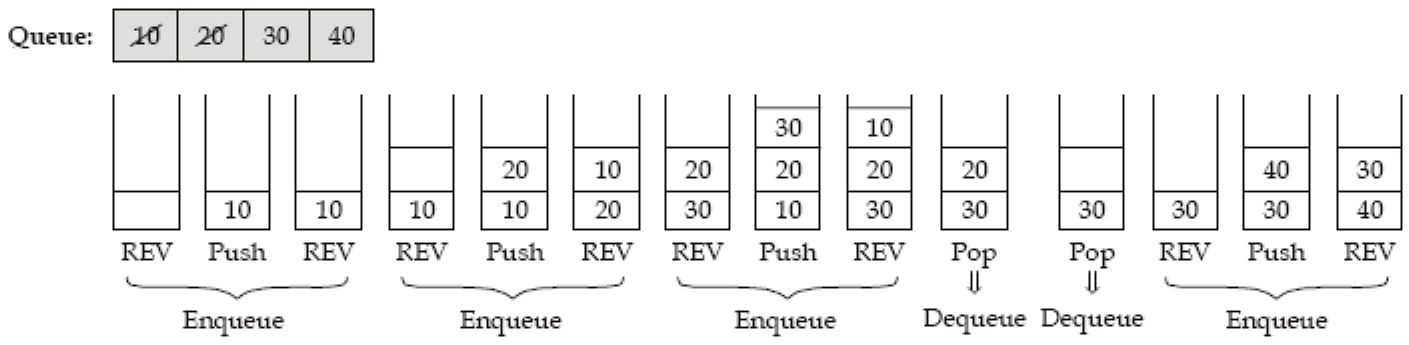
D Either (a) or (b) Correct Option

Solution :
(d)

Enqueue: PUSH ⇒ 1 operation
Dequeue: REVERSE, POP, REVERSE ⇒ 3 operation
Example: Enqueue (10), Enqueue (20), Enqueue (30)
Dequeue, Dequeue, Enqueue (40)



Enqueue: REV, PUSH, REV
Dequeue: POP
Example: Enqueue (10), Enqueue (20), Enqueue (30)
Dequeue, Dequeue, Enqueue (40)



So, either 1 Enqueue and 3 Dequeue or 3 Enqueue and 1 dequeue operation possible.

QUESTION ANALYTICS +

Q. 12

Consider the following C program:

```
int x = 10;
Void Part1(int *a) {
    *a + = x ++;
    printf(“%d” *a);
}
Void Part2(int *b) {
    static x = 15;
    *b = *b × x;
    Part1(&x);
    printf(“%d” *x);
}
Void main ( ) {
    Part2(&x);
    Part1(&x);
}
```

What will be the output using static scoping and dynamic scoping respectively?

Have any Doubt ?

A Static: 165, 303, 303
Dynamic: 31, 301, 301

B Static: 165, 165, 165
Dynamic: 31, 31, 31

C Static: 303, 303, 303
Dynamic: 301, 301, 301

D Static: 165, 165, 303 Your answer is **Correct**

Dynamic: 31, 31, 301

Solution :
(d)

1. Call by reference using static scoping

x $\boxed{\cancel{10}}$ ~~150~~ ~~151~~ ~~302~~ 303 Global variable
1000

Part 2 (1000)

x $\boxed{15}$ Local variable
2000

$*b$ (* 1000)
= 10 × 15
= 150

Part 1 (2000)
$*a$ (*2000)
= 15 + (150 ++)
= 165
print (165)

---

print (165)
Part 1 (1000)
$*a$ (*1000)
= 151 + (151 ++)
= 302
print (303)
"165, 165, 303"

2. Call by reference using dynamic scoping:

x $\boxed{\cancel{10}}$ ~~150~~ ~~300~~ 301
1000

Part 2 (1000)

x $\boxed{\cancel{15}\ \cancel{30}}$ 31
2000

$*b$ = 10 × 15
= 150
Part 1 (2000)
$*a$ = 15 + (15 ++)
= 30
print (31)
print (31)

Part 1 (1000)
$*a$ = 150 + (150 ++)
= 300
print (301)
"31, 31, 301"

QUESTION ANALYTICS +

Q. 13

Which of the following is true?

FAQ Have any Doubt ?

A In sorted array of 'n' distinct elements, deletion of an element take O(log n) time

B In sorted array of 'n' distinct elements, insertion of an element take O(log n) time.

C In sorted array of 'n' distinct elements, finding i^{th} largest element take O(1) time. Your answer is **Correct**

Solution :
(c)

- In sorted array, insertion of an element at beginning take O(n) time, deletion of an element from beginning take O(n) time.
- In sorted array of n elements, finding i^{th} largest or smallest element take O(1) time.
- In unsorted array of n elements insertion of in an array take O(1) time.

D In unsorted array of 'n' distinct elements, insertion of an element take Ω(log n) time..

QUESTION ANALYTICS +

Q. 14

Consider the following C function, where size represent number of elements in an array:

```
int Random (int a[ ], int size) {
    int max_1 = 0,  min_1 = 0, max_2 = 0, start = 0, end = 0, s = 0;
for (int i = 0; i < size; i++) {
    max_2 = max_2 + a[i];
    if (max_1 < max_2) {
        max_1 = max_2;
        start = s;
        end = i;
    }
    if (max_2 < 0) {
        max_2 = 0;
        s = i + 1;
    }
}
return max_1;
}
```

The output return by above function "Random" is ________.

FAQ Have any Doubt ?

A Size of maximum possible sum of array

B Size of largest sum of contiguous sub-array Correct Option

Solution :
(b)

Consider Random array $a[\ ] = \{1, -2, 1, 1, -2, 1\}$
Output is 2 i.e. $\{1, 1\} = 2$
Consider Random array $a[\ ] = \{-2, -3, 4, -1, -2, 1, 5\}$
Output is 7 i.e. $\{4, -1, -2, 1, 5\} = 7$
i.e. sum of largest sum of contiguous sub array.

C Maximum element in any sub-array $a[\ ]$

D Sum of all the elements in the array $a[\ ]$

QUESTION ANALYTICS +

Q. 15

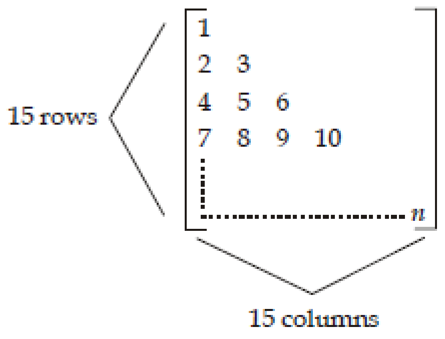
In a lower triangular matrices (size 15 × 15) representation of compact single dimensional array, non-zero elements (i.e. elements of the lower triangle) of each row are stored one after another, starting from the first row. Assume each integer take 1B. The array stored in row major order and first element of array is stored at location 1000, then the address of element $a[10]$ [6] is ________ B.
[**Note:** Only lower triangular elements of the matrix are stored in contiguous array]

FAQ Have any Doubt ?

1061 Correct Option

Solution :
1061

Consider lower triangular matrix:



$$\text{Location } [a[i][j]] = \text{Base address} + \left[\frac{(i-lb_1)(i-lb_1+1)}{2} + (i-lb_2)\right] \times \text{Size of integers}$$

$$= 1000 + \left[\frac{(10-0)(10-0+1)}{2} + (6-0)\right] \times 1 \text{ B}$$

$$= 1000 + \left[\frac{10 \times 11}{2} + 6\right] \times 1 \text{ B}$$

$$= 1000 + [55 + 6] \times 1 \text{ B}$$
$$= 1000 + [61] \times 1 \text{ B}$$
$$= 1000 + 61 \text{ B}$$
$$= 1061 \text{ B}$$

QUESTION ANALYTICS +

Q. 16

An implementation of a queue Q, using two stacks S_1 and S_2, is given below:

```
void enqueue(Q, x) {
    push(S_1, x);
}
void dequeue(Q, x) {
    if (stack-empty(S_2)) then
        if (stack-empty(S_1)) then {
            print("Q is empty");
            return;
        }
    else while (! stack-empty(S_1)) {
        x = pop(S_1);
        push(S_2, x);
    }
    x = pop(S_2);
}
```

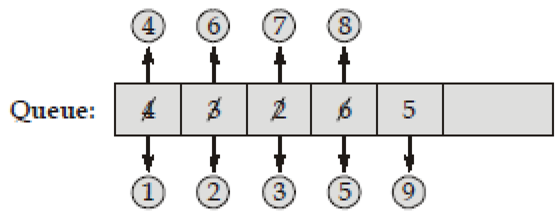
The number Push and Pop operation needed is represented by X and Y, then the value of X + Y for following operation are ________.

Enqueue (4), Enqueue (3), Enqueue (2), Dequeue,
Enqueue (6), Dequeue, Dequeue, Dequeue, Enqueue (5)

FAQ Have any Doubt ?

17 Correct Option

Solution :
17



1. Enqueue (4) = Push (S_1, 4)
2. Enqueue (3) = Push (S_1, 3)
3. Enqueue (2) = Push (S_1, 2)
4. Dequeue = Push (S_2, Pop (S_1)), Push (S_2, Pop (S_1)), Push (S_2, Pop (S_1)), Pop (S_2)
5. Enqueue (6) = Push (S_1, 6)
6. Dequeue (3) = Pop (S_2)
7. Dequeue (2) = Pop (S_2)
8. Dequeue (6) = Push (S_2, Pop (S_1)), Pop (S_2)
9. Enqueue (5) = Push (S_1, 5)

So,   X = Push = 9

                   Y = Pop = 8

So,   X + Y = 17

QUESTION ANALYTICS +

Q. 17

Consider the following C-program:

```
#include <stdio.h>
int value (int *x) {
      static int count;
while (*x) {
      count = count + *x & 1;
      *x >> = 1;
}
return count;
}
int main ( ) {
}
      int a[ ] = {3, 5, 6, 4};
      int y = 0, z = 0;
      for (; y < size of (a)/size of (int); y++)
           z = a[y] + value (a[y]);
}
```

The value of *z* at the end of program is ________.

FAQ | Have any Doubt ?

● 11 Correct Option

Solution :
11

int *z* ~~0~~ ~~5~~ 9

int count ~~0~~ ~~2~~ ~~4~~ ~~6~~ 7 since static variable by default initialize to '0'.

1. $z = 3 + \text{value}(3)$

   └→ Count number of 1's in binary of 3 i.e. 2 (011)

   $z = 3 + 2 = 5$
2. $z = 5 + \text{value}(5)$

   └→ Count number of 1's in binary of 5 i.e. 2 (101) + old value of count

   $z = 5 + 4 = 9$
3. $z = 6 + \text{value}(6)$

   └→ Count number of 1's in binary of 6 i.e. 2 (110) + old value of count

   $z = 6 + 6 = 12$
4. $z = 4 + \text{value}(4)$

   └→ Count number of 1's in binary of 4 i.e. 1 (100) + old value of count

   $z = 4 + 7 = 11$

QUESTION ANALYTICS +