GATE PSUs
State Engg. Exams

MADE EASY Workbook 2024

Detailed Explanations of Try Yourself *Questions*

Computer Science & IT
Programming
and Data Structures

MADE EASY PUBLICATIONS

1 Programming

Detailed Explanation of Try Yourself Questions

T1 : Solution

[O(n^2)]

A(n)

```
{     for (i = 1 to n)
      {     if (n mod i = = 0)
            {     for (j = 1 to n)
                  printf(j)
            }
      }
}
```

The inner `if` block = O(n/2) = O(n); the outer `for` block = O(n).

Time complexity = O(n) × O(n) = O(n^2)

T2 : Solution

[O(1)]

```
main( )
{     int i = 3;
      switch (i)
      {     default : printf("zero")
            Case 1 : printf("one")
                  break
            Case 2 : printf("two")
                  break
            Case 3 : printf("three")
                  break
      }
}
```

Since i = 3 so switch (3) will go to case 3 and run the program only one time.

So time complexity = O(1).

T3 : Solution

1. Const int *P;
   declare P as pointer to const integer.
2. int * const P;
   declare P as constant pointer to integer

T4 : Solution

(*i*) Char (*(*x ( ))[ ])();
declare x as a function returning pointer to array of pointer to function returning char.

(*ii*) Char (*(*x[3])( ) [5];
declare x as array 3 of pointer to function returning pointer to array 5 of char.

(*iii*) Void (*b*int, void (*f)(int))) (int);
Syntac error

(*iv*) Void (*ptr)(int (*)[2], int(*)(void));
Syntax error

T5 : Solution

(b)

Char \ 0
if (0) ∵∵∵ Printf(% S", a) = Null = 0
So condition false
So answer is else part string is not empty.

T6 : Solution

(a)

Since variable d of integer type is static so memory is allocated to it compile time only and same memory is used every time. Therefore, every time old value of *d* (which is update in previous iteration) is used.

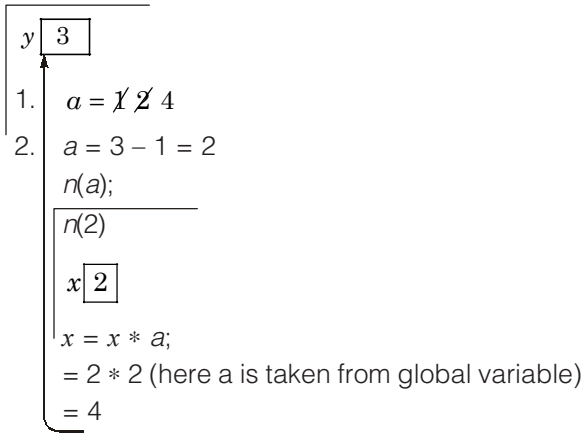
So, output is 312213444.

T7 : Solution

(d)

a 31 globaly initialize.

1. $m(3)$



Printf(4); = 4

3. Printf(a) = 4 since dyanmic scoping is used. So, take value of inner variable 'a'. So answer will be 4, 4.

T8 : Solution

(c)

Take random value of X and Y i.e., $X = 5$ and $Y = 3$.

Initially $X = 5$, $Y = 3$, res = 1, $a = X$ and $b = Y$

Option (a): $X^Y = a^b$

$X^Y = a^b \equiv 5^3 = 5^3 \equiv 125 = 125$

After iteration 1

res = 5; a = 5; b = 2; X = 5; Y = 3

$X^Y = a^b \equiv 5^3 \neq 5^2 \equiv 125 \neq 25$

So, case fail. Option (a) cannot be answer.

Option (b): $(\text{res} * a)^Y = (\text{res} * X)^b$

$(1 \times 5)^3 = (1 \times 5)^3 \equiv 125 = 125$

After iteration 1

res = 5; $a = 5$; $b = 2$; $X = 5$; $Y = 3$

$(\text{res} * a)^Y = (\text{res} * X)^b \equiv (5 \times 5)^3 = (5 \times 5)^2$

$15625 \neq 625$ So, case fail. Option (b) cannot be answer.

Option (d): $X^Y = (\text{res} * a)^b$

$5^3 = (1 \times 5)^3 \equiv 125 = 125$

After iteration 1

res = 5; $a = 5$; $b = 2$; $X = 5$; $Y = 3$

$X^Y = (\text{res} * a)^b \equiv 5^3 = (5 \times 5)^2$

$125 \neq 625$ So, case fail.

Option (d) cannot be answer.

Option (c):

$X^Y = \text{res} * a^b$

$5^3 = 1 \times 5^3 \equiv 125 = 125$

After iteration 1

$\text{res} = 5;\ a = 5;\ b = 2;\ X = 5;\ Y = 3$

$X^Y = \text{res} * a^b \equiv 5^3 = 5 \times 5^2 \equiv 125 = 125$

After iteration 2

$\text{res} = 25;\ a = 5;\ b = 1;\ X = 5;\ Y = 3$

$X^Y = \text{res} * a^b \equiv 5^3 = 25 \times 5^1 \equiv 125 = 125$

So, all cases are passes.

So option (c) will the answer.

T9 : Solution

a	3	5	2	6	4

1. $f(a, 5)$ is a function contain 2 parameter one contain starting address of array and second parameter tell number of element in the array.
2. Every time 'n' value compare with 1 when it is less than equal to 1 return 0 and stop the program otherwise continue with recursive function call.

1. $f(a, 5)$
   $*P = a$; P pointed to same address pointed by 'a'.
   $n = 5$; n value greater than 1.
   So, max ($f(P + 1, 5 - 1), 3 - 5$); or
   max ($f(P + 1, 4) - 2$);
2. $f(P + 1, 4)$
   $*P = P + 1$; P is pointed to next element of array i.e., 5.
   $n = 4$; n value greater than 1.
   So, max (max($f(P + 1, 4 - 1), 5 - 2), -2$) or
   max (max($f(P + 1)$ 3), 3), –2)
3. $f(P + 1, 3)$
   $*P = P + 1$; P is pointed to next element of array i.e., 2.
   $n = 3$; n value greater than 1.
   So, max(max(max($f(P + 1, 3 - 1), 2 - 6), 3) -2$) or max(max(max($f(P + 1, 2), - 4), 3) -2$);
4. $f(P + 1, 2)$;
   $*P = P + 1$; P is pointed to next element of array i.e., 6.
   $n = 2$; n value greater than 1.
   So, max(max(max(max($f(P+ 1, 2 - 1), 2),-4), 3), -2$) or
   max(max(max(max($f(P+1, 1), 2), -4), 3),-2$)
5. $f(P + 1, 1)$;
   $*P = P+1$; P is pointed to next element of array i.e., 4.
   $n = 1$; n value equal to 1 so, return 0.
   So max(max(max(max(0, 2), –4), 3), –2)
   max(max(max(2, –4), 3), –2)
   max(max(2, 3), –2)
   max(3, –2) = 3

So the value printed by given code is 3.

■■■■

2 Linked List, Stack, Queue and Hashing

T1 : Solution

Implementation of stack using single linklist:

Inserting sequence: 1, 2, 3, 4, 5, 6

Insertion take 0(1) time

Link list representation:

1. → [1 | /]
2. → [2 |] → [1 | /]
3. → [3 |] → [2 |] → [1 | /]
4.
5.
6. Insertion takes 0(1) time.

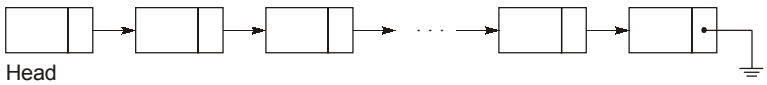
Deletion in stack (Pop)

Remove top element every time so 0(1)

Deletion in linklist

Remove 1st node every time with making second node to head.

T2 : Solution



enqueue operation takes $O(1)$ time

dequeue operation takes $O(n)$ time [visits last node]

T3 : Solution

(d)

PUSH (S, P, Q, T_i, x)
{
    if $\left(T_i == \left(\frac{P}{Q} \times (i+1) - 1\right)\right)$
    {
        printf ("stack overflow");
        exit (1);
    }
    else
        T_i++;
    $S[T_i] = x$;
}

$T_i == \left(\frac{P}{Q} \times (i+1) - 1\right)$ indicate the last location of the array is already filled. So overflow occur.

T4 : Solution

(a)

Number of push operations = n(insert) + m(delete) = $n + m$

So, $n + m \le x$ but there are maximum $2n$ insert operations so $n + m \le x \le 2n$ ...(1)

Number of pop operations = $n + m$

But there are 2m delete operations which are less than no. of pop operations, hence

$$2m \le n + m \quad \text{...(2)}$$

From (1) and (2): $n + m \le x \le 2n$ and $2m \le n + m$

T5 : Solution

(22079)

Formula to find location of $a[20][20][30] = 10 + \{[(20-1)(30-1)(40-1)] + (20-1)(30-1) + (30-1)\}$

$= [10 + (19 \times 29 \times 39) + (19 \times 29) + (29)]$

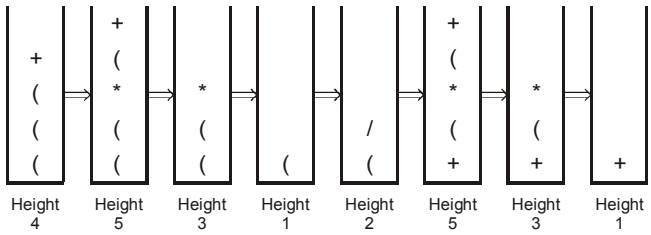
$= 10 + 21489 + 551 + 29$

$= 10 + 22069$

$= 22079$

T6 : Solution

(5)



(Uniqueue heights are 1, 2, 3, 4, 5) whered 1, 3, 5 are repeated two times each. Maximum size of stack is 5.

T7 : Solution

(0.7324)

Expected number of probes in a unsuccessful = $\frac{1}{(1-\alpha)}$

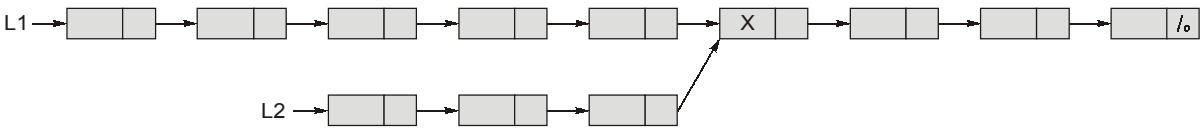
$$\frac{1}{1-\alpha} = 3$$
$$1 = 3(1-\alpha)$$
$$1 = 3 - 3\alpha$$
$$-2 = -3\alpha$$
$$\alpha = \frac{2}{3}$$

Expected number of probes in a unsuccessful = $1/\alpha \log_e 1/(1-\alpha)$

$$\frac{3}{2}\log_e 3 = 0.7324$$

T8 : Solution

(b)



We need to traverse both the linked list of size m and n.

So it will take $O(m + n)$.

T9 : Solution

(b)

By using BSF (Breadth First Search) traversal we can set the twin pointer in each entry in each adjacency list.

So it will take $\Theta(m + n)$ times (since adjacency list are using).

