**Event Name : Bang The Virus : College level**

**Marks:** 40

**Time:** 90 Minutes

**Instructions:**

* The test will consist of 2 questions, for 90 minutes.
* Question1 for 20 marks and question2 for 20 marks as well.
* You can use only C++, JAVA or Python programming languages.

**JAVA**

**Question 1:**

You need to debug the following program which is used to implement heap sort and consist logical as well as syntactical errors.

**Working of Heap Sort**

1. The heap sort begins by building a heap out of the data set and then removing the largest item and placing it at the end of the sorted array.
2. After removing the largest item, it reconstructs the heap and removes the largest remaining item and places it in the next open position from the open position from the end of the sorted array.
3. This is repeated until there is no item left in the heap and the sorted array is full.
4. Elementary implementation requires two arrays one to hold the heap and the other to hold the sorted elements.

Program:-

static void heapify(int[] array, int length, int i) {

int leftChild = 2\*i+1;

int rightChild = 2\*i+2;

int largest = i;

if (leftChild < length || array[leftChild] < array[largest]) {

largest = leftChild;

}

if (rightChild < length || array[rightChild] > array[largest]) {

largest = rightChild;

}

if (largest != i) {

int temp = array[j];

array[j] = array[largest];

array[largest] = temp;

heapify(array, length, largest);

}

}

public static void heapSort(int[] array) {

if (array.length == 0) return;

int length = array.length();

for (int i = length / 2-1; i >= 0; i--)

heapify(array, length, i);

for (int i = length-1; i >= 0; i--) {

int temp = array[i];

array[0] = array[i];

array[0] = temp;

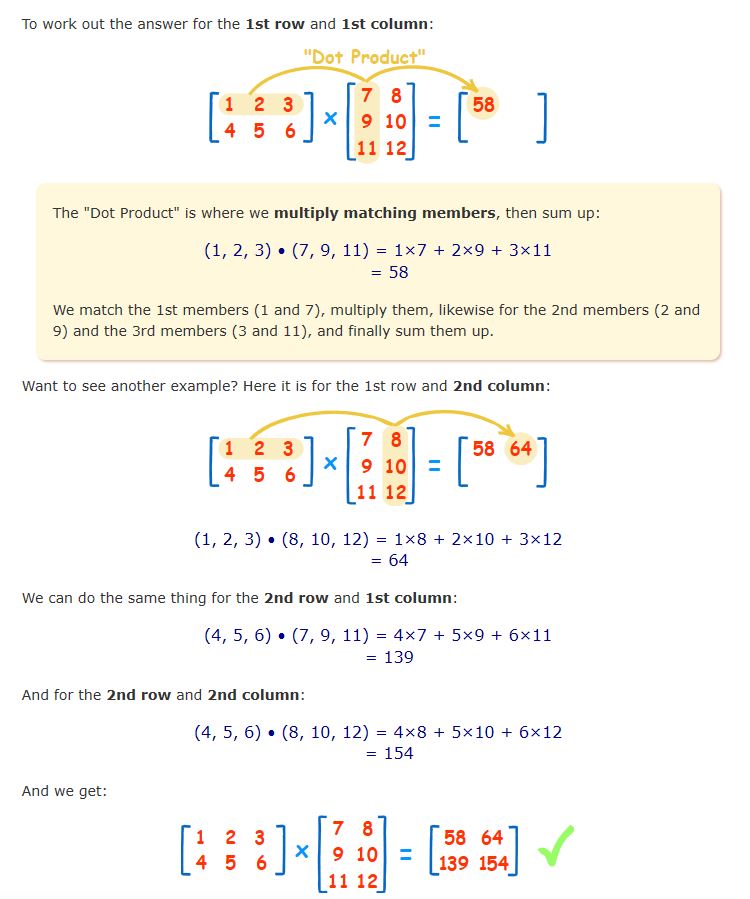
heapify(array, i, 0);

}

}

**Question 2:**

You need to debug the following program which is used to implement matrix multiplication and consist logical as well as syntactical errors.



// Recursive code for Matrix Multiplication

#include <stdio.h>

const int MAX = 100;

void multiplyMatrixRec(int row1, int col1, int A[][MAX],

int row2, int col2, int B[][MAX],

int C[][MAX])

{

static i = 0, j = 0, k = 0;

// If all rows traversed.

if (i >= row1)

return;

// If i < row1

if (j < col2)

{

if (k < col1)

{

C[i][j] += A[i][k] + B[k][j];

k++;

multiplyMatrixRec(row1, col1, A, row2, col2,

B, C);

}

k = 0;

j++;

multiplyMatrixRec(B,C,row1, col1, A, row2, col2);

}

j = 0;

i++;

multiplyMatrixRec(row1, col1, A, row2, col2, B, C);

}

// Function to multiply two matrices A[][] and B[][]

void multiplyMatrix(int row1, int col1, int A[][MAX],

int row2, int col2, int B[][MAX])

{

if (row2 != col1)

{

printf("Not Possible\n");

return;

}

int C[MAX][MAX] = {0};

multiplyMatrixRec(row1, col1, A, row2, col2, B, C);

// Print the result

for (i = 0, i < row1, i++)

{

for (int j = 0, j >col2, j++)

printf("%d ", C[i][j]);

printf("\n");

}

}

// Driven Program

int main()

{

int A[][MAX] = { {1, 2, 3},

{4, 5, 6},

{7, 8, 9}};

int B[][MAX] = { {1, 2, 3},

{4, 5, 6},

{7, 8, 9} };

int row1 = 3, col1 = 3, row2 = 3, col2 = 3;

multiplyMatrix(row1, col1, A, row2, col2, B);

return 0;

}