**Max Points**: **60**

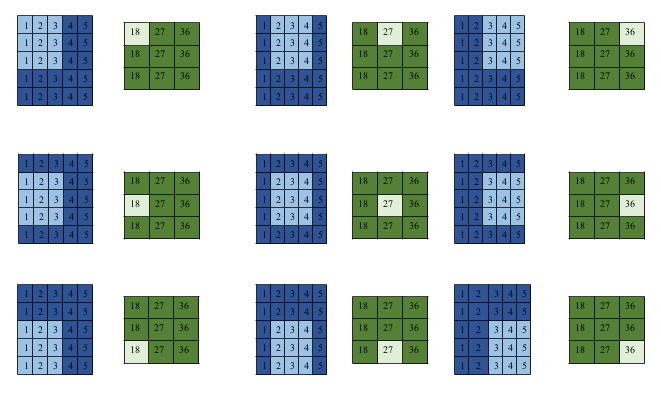
**Question#1:** Give the output when these programs are executed: **[12 points, CLO3, 15 mins]**

|  |  |
| --- | --- |
| **#include<stdio.h>**  **int main()**  **{**  **int i = 1;**  **do**  **{**  **while(i)**  **i--;**  **for(i++; 0; i++);**  **break;**  **}while(1);**  **printf("%d", i);**  **return 0;**  **}** | **#include<stdio.h>**  **int main()**  **{**  **alpha[0] = 5;**  **for (count = 1; count < 5; count++)**  **{**  **alpha[count] = 5 \* count + 10;**  **alpha[count - 1] = alpha[count] - 4;**  **}**  **for (count = 0; count < 5; count++){**  **printf("%d", alpha[count] );**  **}**  **}** |
| **#include<stdio.h>**  **int main()**  **{**  **int a[5] = {5, 1, 15, 20, 25};**  **int i, j, m;**  **i = ++a[1];**  **j = a[1]++;**  **m = a[i++];**  **printf("%d, %d, %d", i, j, m);**  **return 0;**  **}** | **#include<stdio.h>**  **int main()**  **{**  **int i = 0;**  **while(++i)**  **{**  **i == --i?i = 0:i = 1;**  **}**  **printf("%d", i);**  **return 0;**  **}** |

**Q2 Scenario:** A **convolution** is a type of matrix operation, consisting of a kernel, a small matrix of weights, that slides over input data performing element-wise multiplication with the part of the input it is on, then summing the results into an output.

You need to scan the arrays **int input [5][5]** and **int filter [3][3]** and declare the array **int result [3][3]** to store the result of convolution.

The example of convolution operation is mentioned below. Note in this case the **filter** is initialized with 1. Furthermore, **green matrix** represents result, **blue matrix** represents input and **light blue** matrix represents filter (which keeps on moving).



***Q3 Yasir*** and ***Binish*** are playing a game, where ***yellow*** or ***blue*** pieces are represented by a colour string. The game rules are as follows:

* ***Yasir*** moves first then they take alternate turns
* With each move, ***Yasir*** may remove a ***yellow*** piece that has adjacent ***yellow*** pieces on both sides
* Likewise, with each move, ***Binish*** may remove any ***blue*** piece that has adjacent ***blue*** pieces on both sides.
* After a piece is remove the string is reduce in size by one piece. For example, removing ***‘B’*** from ***“ABC”*** results in ***“AC”***.
* When a player can no longer move, they have lost the game.

**Example:** Colours = ***“yyybbbbyyy”***

***Yasir*** removes the piece ***‘y’*** at index 1, colours = ***“yybbbbyyy”***

***Binish*** removes the piece ***‘b’*** at index 3, colours = ***“yybbbyyy”***

***Yasir*** removes the piece ***‘y’*** at index 6, colours = ***“yybbbyy”***

***Binish*** removes the piece ***‘b’*** from index 3, colours = ***“yybbyy”***

***Yasir*** has no other moves, so ***Binish*** wins. Display ***Binish***!

Determine who wins if ***Yasir*** and ***Binish*** both play with optimum skill. Display the string ‘***Yasir***’ or ‘***Binish***’.