Part 1:

Mat1:

1. #Define Larsen = 6 & Jacob = 5
2. Nested for loop using Larsen & Jacob as <parameters
3. Run normal nested for loop with index+=1 after each pass
4. Arr[col][row] = index

Mat2:

1. Nested for loop with index += 3 starting at 2
2. Switch Larsen and Jacob parameters in nested for loop
3. Swap col/row parameters for array (arr[col][row])
4. Arr[col][row] = index

Mat3:

1. Same as mat1 using 2 and 4 instead of Larsen and Jacob and have index start at ten and -= 2 after each pass

Mat4:

1. Same as Mat2 using 4 and 2 instead and have index start at -6 and += 1.5 per pass

Part2:

1. global arrays of [6][5]
2. Ask for user input twice assign to 2 ints
3. Use user ints to open correct files
4. Define sizex&y based on which file is opened
5. Make sizex&y of both matrixes are the same
6. If yes add them together
7. If no stop program
8. Show output array and send it to output file

Part3:

1. Same as part 2 up until size check
2. Check if y1 == x2
3. If yes triple nested for loop with loop 1 < sizeX1 2 < sizeY2 3 < sizeY1
4. Multiply whole row by whole col based on loop 1 & 2 index
5. Input multiple into new matrix
6. Cout and send to file each new matrix index

Part 4:

1. Same as part 3 up until after y1==x2 check
2. Use library to multiply
3. Send output to file

Part 5:

1. Ask for inputs
2. Open files according to inputs
3. 2 vectors assigned numbers from 2 files
4. Output double is vector1[0]\*vector2[0] + vector1[1]+vector2[1]
5. Send output to file

Part 6:

1. Take input
2. Find sizex&y based on file input
3. Make new matrix of same/opposite size ([5][6]->[6][5])
4. Nested for loop matrix2[i][j]=matrix1[j][i]
5. Send output to file

Part 7:

1. Copy code from parts 5 and 6 and have the library do all the calculations
2. Send output(s) to file