Homework Assignment Template

Your Name

June 10, 2025

Homework Assignment

Question 1: Basic Arithmetic Function

Implement a function that calculates the sum of squares of numbers from 1 to n.

Solution

- **Python Code:**

```python
def sum_of_squares(n):
total = 0
for i in range(1, n + 1):
total += i ** 2
return total

```

- **MATLAB Code:**

```matlab
function result = sumOfSquares(n)
result = sum((1:n).^2);
end

```

- **C Code:**

```c
int sumOfSquares(int n) {
    int total = 0;
    for (int i = 1; i <= n; i++) {
        total += i * i;
    }
    return total;
}

```

Question 2: Matrix Operations

Write a program to compute the transpose of a given matrix.

Solution

- **Python Code:**

```python
def transpose_matrix(matrix):
return [[matrix[j][i] for j in range(len(matrix))] for i in
  range(len(matrix[0]))]

```

- **MATLAB Code:**

```matlab
function result = transposeMatrix(matrix)
result = matrix';
end

```

- **C Code:**

```c
void transposeMatrix(int rows, int cols, int matrix[rows][cols], int
  result[cols][rows]) {
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            result[j][i] = matrix[i][j];
        }
    }
}

```

Question 3: Data Analysis

Create a table to summarize the performance metrics of an algorithm (e.g., time and accuracy).

Solution

Below is a sample table summarizing performance metrics:

Table 1: Algorithm Performance Metrics

Algorithm	Execution Time (s)	Accuracy (%)
Algorithm A	0.25	95.3
Algorithm B	0.30	92.1
Algorithm C	0.18	97.2

Question 4: Image Processing

Describe an algorithm to process two images side by side.

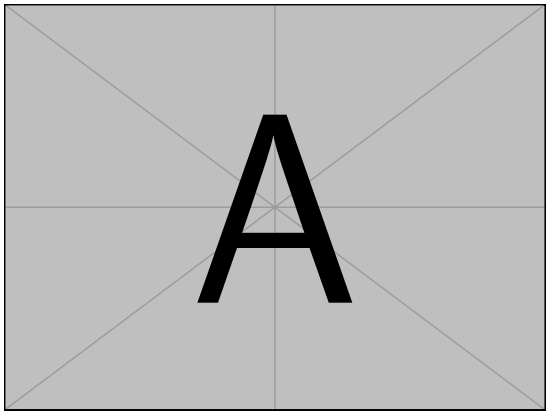
Solution

Below are two sample images displayed side by side:

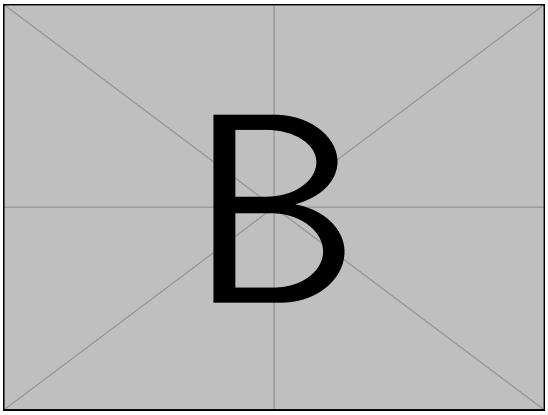
- **Python Code (Image Processing Placeholder):**

```python
import cv2
def process_image(image):
# Placeholder for image processing
return cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

```



(a) Image A



(b) Image B

Figure 1: Side-by-side comparison of two images.

Question 5: Sorting Algorithm

Implement a sorting algorithm and provide its pseudocode.

Solution

Below is the pseudocode for QuickSort:

Algorithm 1 QuickSort Algorithm

```
procedure QuickSort(array, low, high)
    if low < high then
        pivotIndex ← Partition(array, low, high)
        QuickSort(array, low, pivotIndex - 1)
        QuickSort(array, pivotIndex + 1, high)
    end if
end procedure
procedure Partition(array, low, high)
    pivot ← array[high]
    i ← low - 1
    for j ← low to high - 1 do
        if array[j] ≤ pivot then
            i ← i + 1
            Swap array[i] and array[j]
        end if
    end for
    Swap array[i využívané + 1] and array[high]
    return i + 1
end procedure
```

- **Python Code:**

```python
def quicksort(arr, low, high):
if low < high:
pi = partition(arr, low, high)
quicksort(arr, low, pi - 1)
quicksort(arr, pi + 1, high)

```

```
def partition(arr, low, high):
pivot = arr[high]
i = low - 1
for j in range(low, high):
if arr[j] <= pivot:
i += 1
arr[i], arr[j] = arr[j], arr[i]
arr[i + 1], arr[high] = arr[high], arr[i + 1]
return i + 1

```