LABWORK 5: Longest Path

Student's name: Hoang Quoc Minh Quan
Student's ID: BI12-363

I. Workflow

a. Mapper

The mapper function reads input from a file and maps each line (representing a file path) to a key-value pair. The key is the length of the path, and the value is the path itself.

1. **Read Input:** Using "fgets", the function reads lines (file paths) from the provided file ("input").

```
while (fgets(line, sizeof(line), input) != NULL)
```

2. **Process Each Line:**
   - Remove any newline character from the end of the line using "strlen" and "'\n'" check.

```
// Remove newline character if present
size_t len = strlen(line);
if (len > 0 && line[len - 1] == '\n')
{
    line[len - 1] = '\0';
}
```

   - Determine the length of the file path using "strlen".

```
// Calculate path length
size_t path_length = strlen(line);
```

   - Store the path length as the key and the path itself as the value in the "output" array.

```
// Emit key-value pair
output[*num_items].key = path_length;
strcpy(output[*num_items].value, line);
(*num_items)++;
```

3. **Track Number of Items:** Increment the “num_items” counter for each successfully processed line.

```
// Check if we exceed the maximum number of items
if (*num_items >= MAX_ITEMS)
{
```

4. **Check Limit:** If the number of processed items exceeds the maximum (“MAX_ITEMS”), an error message is printed, and processing stops.

```
fprintf(stderr, "Exceeded maximum number of items\n");
break;
```

b. Reducer

The reducer function takes the mapped key-value pairs (file paths and their lengths) and identifies the longest path among them.

1. **Initialize Variables:**
   - “longest_path”: String to store the longest path encountered.
   - “max_length”: Variable to track the maximum path length.

```
void reduce(struct KeyValue *input, int num_items)
{
    char longest_path[MAX_PATH_LENGTH];
    size_t max_length = 0;
```

2. **Iterate Over Items:**
   - Loop through each key-value pair in the “input” array.

```
for (int i = 0; i < num_items; i++)
{
    size_t length = input[i].key;
    char *path = input[i].value;
```

   - For each pair, compare the length (“length”) with “max_length”:
   - If “length” is greater than “max_length”, update “max_length” and copy the path (“path”) to “longest_path”.

```
// Check if this path has the longest length encountered so far
if (length > max_length)
{
    max_length = length;
    strcpy(longest_path, path);
}
```

   - If “length” is equal to “max_length”, print the path (handles multiple longest paths).

```
else if (length == max_length)
{
    // If same length, output this path (for multiple longest paths)
    printf("%s\n", path);
}
```

3. **Output Results:**
   - After processing all items, print the identified longest path (“longest_path”) along with a message indicating it's the longest path found.

```
// Output the longest path found
printf("Longest Path(s):\n%s\n", longest_path);
```

c. *Figure*

Here’s is the figure that illustrated the workflow of Mapper and Reducer:

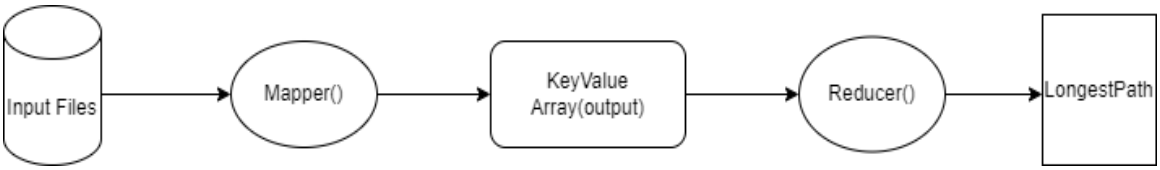
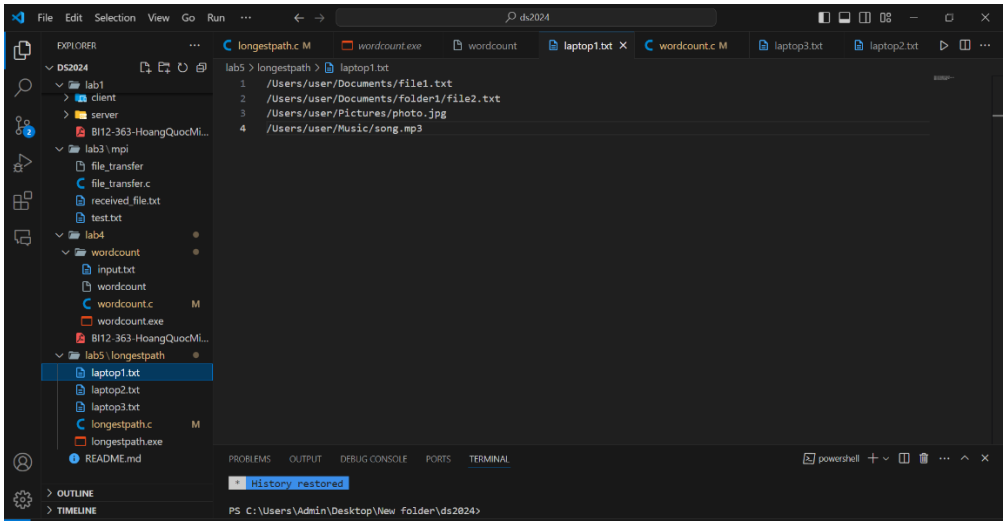
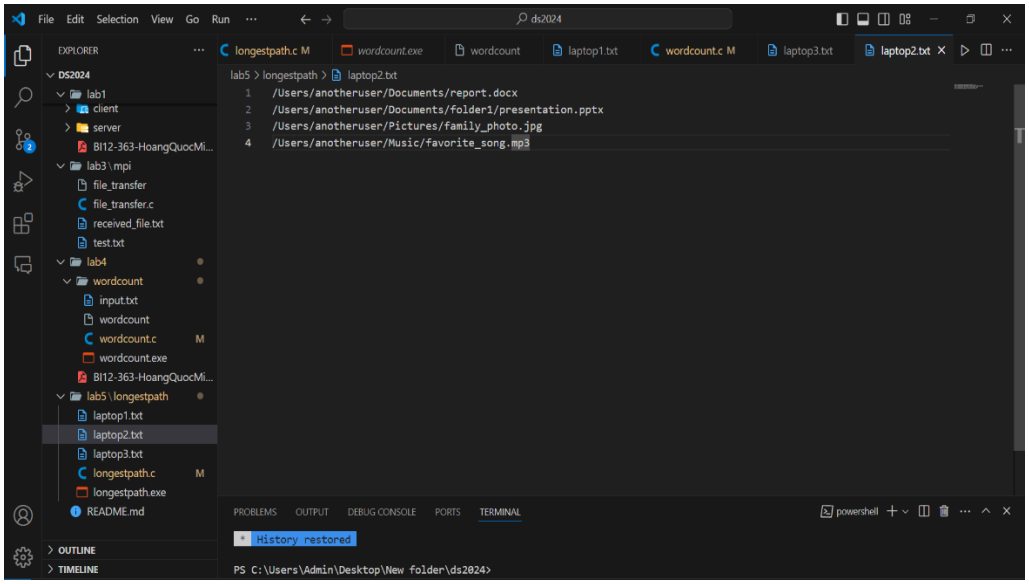


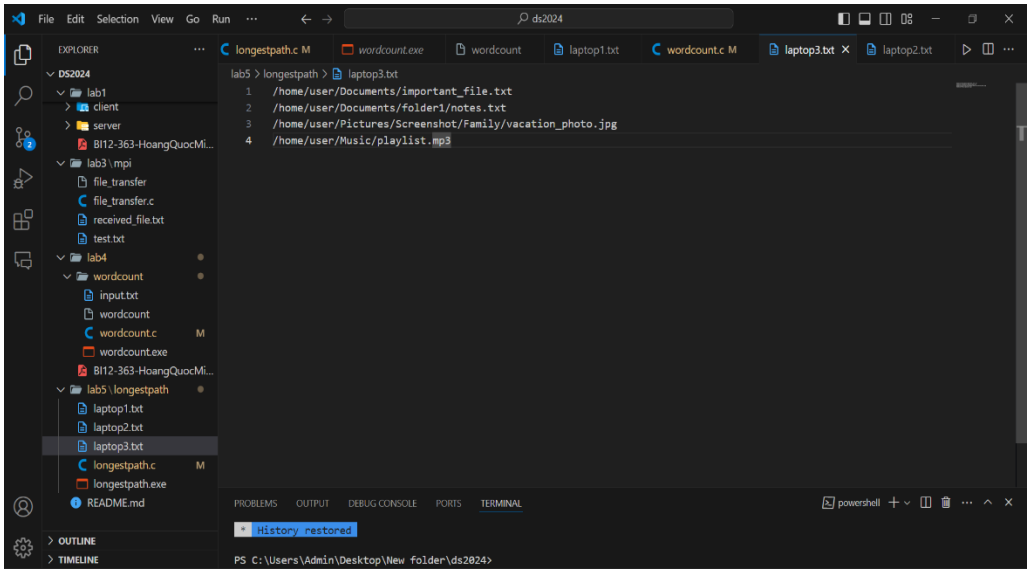
Figure 1. Workflow of MapReduce

II. Implementation

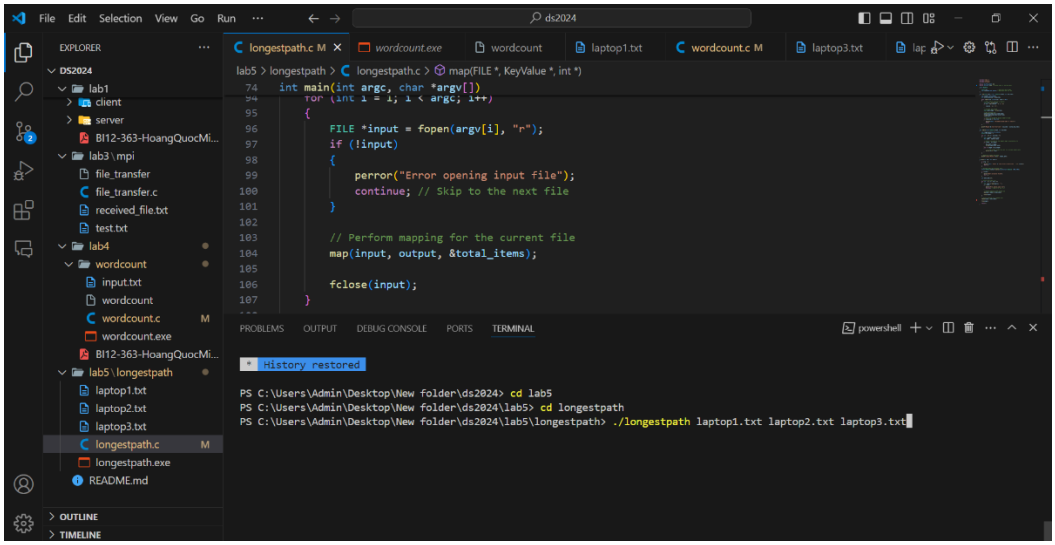
These are 3 test files, which contains the example paths:







Now we run the code with 3 example input files



This is the result: