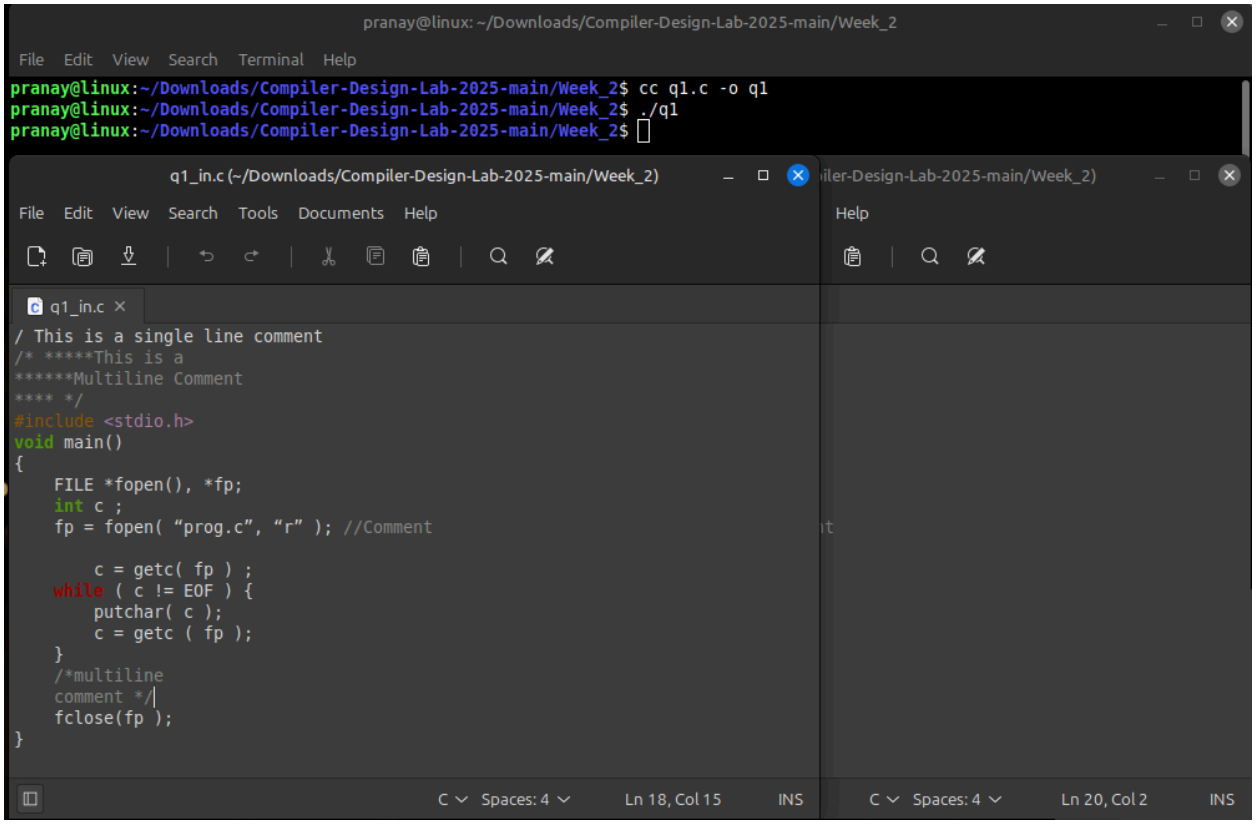
CD LAB - WEEK 2

Pranay Goel
220905524
CSE B

Write a ‘C’ program

1. that takes a file as input and replaces blank spaces and tabs by single space and writes the output to a file

```
#include<stdio.h>
#include<stdlib.h>

int main(){
	FILE *fa, *fb;
	int c;
	fa = fopen("q1_in.c", "r");
	if (fa == NULL) {
		printf("Cannot open file \n");
		exit(0);
	}
	fb = fopen("q1_out.c", "w");

	c = getc(fa);
	while (c != EOF) {
		if(c == '\t' || c == ' ') {
			putc(' ', fb);
			while(c == ' ' || c == '\t')
				c = getc(fa);
			putc(c, fb);
		}
		else putc(c, fb);
		c = getc(fa);
	}
	fclose(fa);
	fclose(fb);
	return 0;
}
```

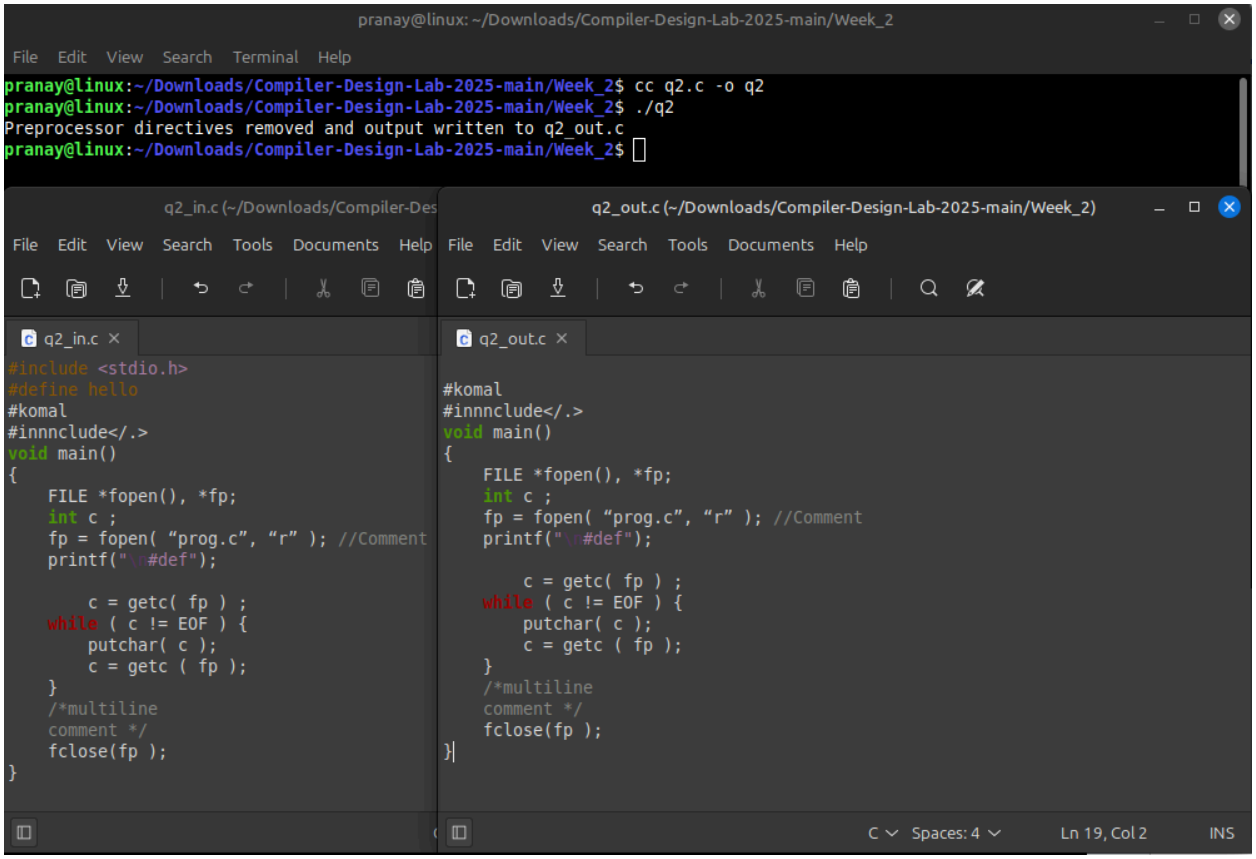


2. to discard preprocessor directives from the given input 'C' file.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_BUFFER_SIZE 1024

int isPreprocessorDirective(const char *line) {
    if (strncmp(line, "#include", 8) == 0 ||
        strncmp(line, "#define", 7) == 0 ||
        strncmp(line, "#ifdef", 6) == 0 ||
        strncmp(line, "#ifndef", 7) == 0 ||
        strncmp(line, "#endif", 6) == 0) {
        return 1;
    }
    return 0;
}


int main(int argc, char *argv[]) {
    FILE *fa, *fb;
```

```
char c;
char buffer[MAX_BUFFER_SIZE];
int bufferIndex = 0;

fa = fopen("q2_in.c", "r");
if (fa == NULL) {
    printf("Error opening input file.\n");
    return 1;
}

fb = fopen("q2_out.c", "w");
if (fb == NULL) {
    printf("Error opening output file.\n");
    fclose(fa);
    return 1;
}

while ((c = getc(fa)) != EOF) {
    if (c == '#'){
        bufferIndex = 0;
        buffer[bufferIndex++] = c;
        while ((c = getc(fa)) != EOF && c != '\n' && bufferIndex < MAX_BUFFER_SIZE - 1){
            buffer[bufferIndex++] = c;
        }
        buffer[bufferIndex] = '\0';  // Null-terminate the buffer

        if (!isPreprocessorDirective(buffer)){
            // If not a directive write the # followed by the content to the output file
            for (int i = 0; i < bufferIndex; i++) {
                putc(buffer[i], fb);
            }
        }

        if (c != EOF && c == '\n'){
            putc(c, fb);
        }
    }
    else{
        putc(c, fb);
    }
}

fclose(fa);
fclose(fb);
```

```
    printf("Preprocessor directives removed and output written to q2_out.c\n");
}
```



3. that takes a C program as input, recognizes all the keywords and prints them in uppercase.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

#define MAX_BUFFER_SIZE 100
#define NUM_KEYWORDS 7

void toUpperCase(char* str) {
    for (int i = 0; str[i] != '\0'; i++) {
        str[i] = toupper(str[i]);
    }
}

int isKeyword(const char* word, char* keywords[], int num_keywords) {
    for (int i = 0; i < num_keywords; i++) {
        if (strcmp(word, keywords[i]) == 0)
```

```
            return 1;
        }
        return 0;
    }

    int main() {
        FILE *fa, *fb;
        char c;
        char buffer[MAX_BUFFER_SIZE];
        int bufferIndex = 0;
        char *keywords[] = {"include", "define", "if", "else", "while", "for", "return"};

        fa = fopen("q3_in.c", "r");
        if (fa == NULL) {
            printf("Error opening input file.\n");
            return 1;
        }

        fb = fopen("q3_out.c", "w");
        if (fb == NULL) {
            printf("Error opening output file.\n");
            fclose(fa);
            return 1;
        }

        while ((c = getc(fa)) != EOF) {
            if (isalpha(c) || c == '_') {
                buffer[bufferIndex++] = c;

                // Make sure the buffer doesn't overflow
                if (bufferIndex >= MAX_BUFFER_SIZE) {
                    bufferIndex = MAX_BUFFER_SIZE - 1;
                }
            }
            else {
                if (bufferIndex > 0) {
                    buffer[bufferIndex] = '\0';
                    if (isKeyword(buffer, keywords, NUM_KEYWORDS)) {
                        toUpperCase(buffer);
                    }
                    for (int i = 0; buffer[i] != '\0'; i++) {
                        putc(buffer[i], fb);
                    }
                    bufferIndex = 0;
```

```
            }
            putc(c, fb);
        }
    }

    // If there's any word left in the buffer ie file ends with word
    if (bufferIndex > 0) {
        buffer[bufferIndex] = '\0';
        if (isKeyword(buffer, keywords, NUM_KEYWORDS)) {
            toUpperCase(buffer);
        }
        for (int i = 0; buffer[i] != '\0'; i++) {
            putc(buffer[i], fb);
        }
    }

    fclose(fa);
    fclose(fb);
    printf("Keywords converted to uppercase and written to q3_out.c\n");
}
```