

Study Time: 5.5 hours

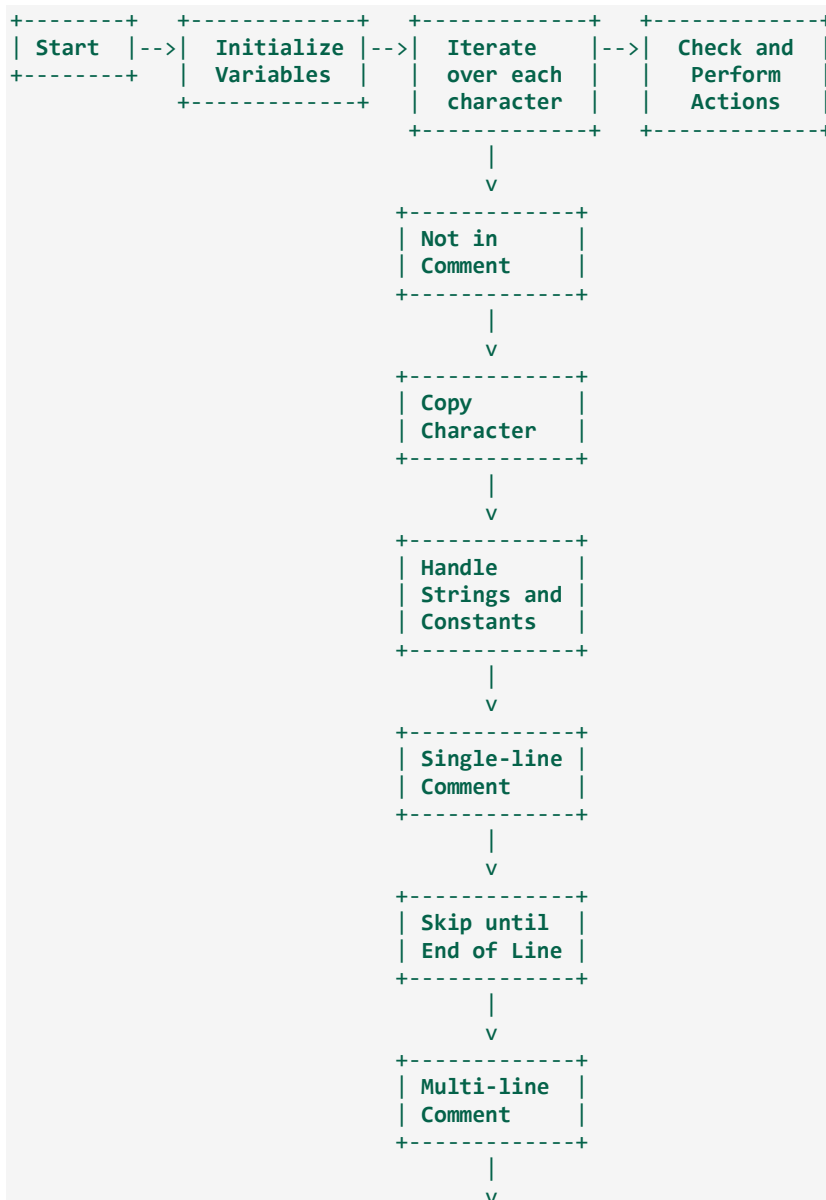
Exercise Time: 2.5 hours

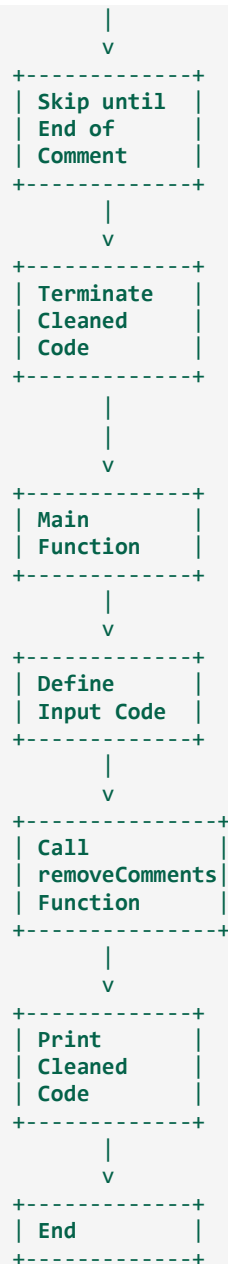
Documentation of Day 24

Exercise 1-23.

Write a program to remove all comments from a C program. Don't forget to handle quoted strings and character constants properly. C comments don't nest.

Below I have given a flowchart representing the logical approach of to implement the program:





Source Code:

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

/*****
/* FUNCTION NAME : isInQuotedString
/* INPUTS      : char* line - Input line, int index - Index to check
/* RETURN     : bool
/* NOTES      : Checks if the character at the given index is within a
/*              quoted string.
*****/
bool isInQuotedString(char* line, int index) {
    bool inString = false;
    int i;
    for (i = 0; i < index; i++) {
        if (line[i] == '\"' && (i == 0 || line[i-1] != '\\')) {
            inString = !inString;
        }
    }
    return inString;
}

/*****
/* FUNCTION NAME : isInCharacterConstant
/* INPUTS      : char* line - Input line, int index - Index to check
/* RETURN     : bool
/* NOTES      : Checks if the character at the given index is within a
/*              character constant.
*****/
bool isInCharacterConstant(char* line, int index) {
    bool inConstant = false;
    int i;
    for (i = 0; i < index; i++) {
        if (line[i] == '\'' && (i == 0 || line[i-1] != '\\')) {
            inConstant = !inConstant;
        }
    }
    return inConstant;
}
```

```

/*****
/* FUNCTION NAME : removeComments
/* INPUTS      : char* line - Input line
/* RETURN     : void
/* NOTES      : Removes comments from the input line. Comments can be
/*              single-line (//) or multi-line (/* ... */).
*****/

void removeComments(char* line) {
    bool inString = false;
    bool inConstant = false;
    bool inSingleLineComment = false;
    bool inMultiLineComment = false;
    char* cleanLine = line;
    int i = 0;
    while (line[i] != '\0') {
        if (!inString && !inConstant && !inMultiLineComment && line[i] == '/' &&
line[i + 1] == '/') {
            inSingleLineComment = true;
            i += 2; // Skip the comment characters
            continue;
        }

        if (!inString && !inConstant && !inSingleLineComment && line[i] == '/' &&
line[i + 1] == '*') {
            inMultiLineComment = true;
            i += 2; // Skip the comment characters
            continue;
        }

        if (inSingleLineComment && line[i] == '\n') {
            inSingleLineComment = false;
        }

        if (inMultiLineComment && line[i] == '*' && line[i + 1] == '/') {
            inMultiLineComment = false;
            i += 2; // Skip the comment characters
            continue;
        }

        if (!inSingleLineComment && !inMultiLineComment) {
            *cleanLine = line[i];
            cleanLine++;
        }
    }
}

```

```

    }

    if (line[i] == '\"' && (i == 0 || line[i-1] != '\\')) {
        inString = !inString;
    }

    if (line[i] == '\'' && (i == 0 || line[i-1] != '\\')) {
        inConstant = !inConstant;
    }

    i++;
}
*cleanLine = '\\0';
}

/*****
/* FUNCTION NAME : main
/* INPUTS      : void
/* RETURN      : int
/* NOTES       : Tests the removeComments function by removing comments
/*              from the given code and printing the resulting code.
*****/

int main() {
    char code[] = "\
#include <stdio.h>\n\
\n\
int main() {\n\
    // This is a comment\n\
    printf(\"Hello, world!\"); /* Another comment */\n\
    /*\n\
    This is a multi-line comment\n\
    printf(\"This line won't be printed\");\n\
    */\n\
    printf(\"This line will be printed\");\n\
    printf(\"This line /* with a nested comment */ will be printed\");\n\
    return 0;\n\
}";

    removeComments(code);
    printf("%s\n", code);

    return 0;
}

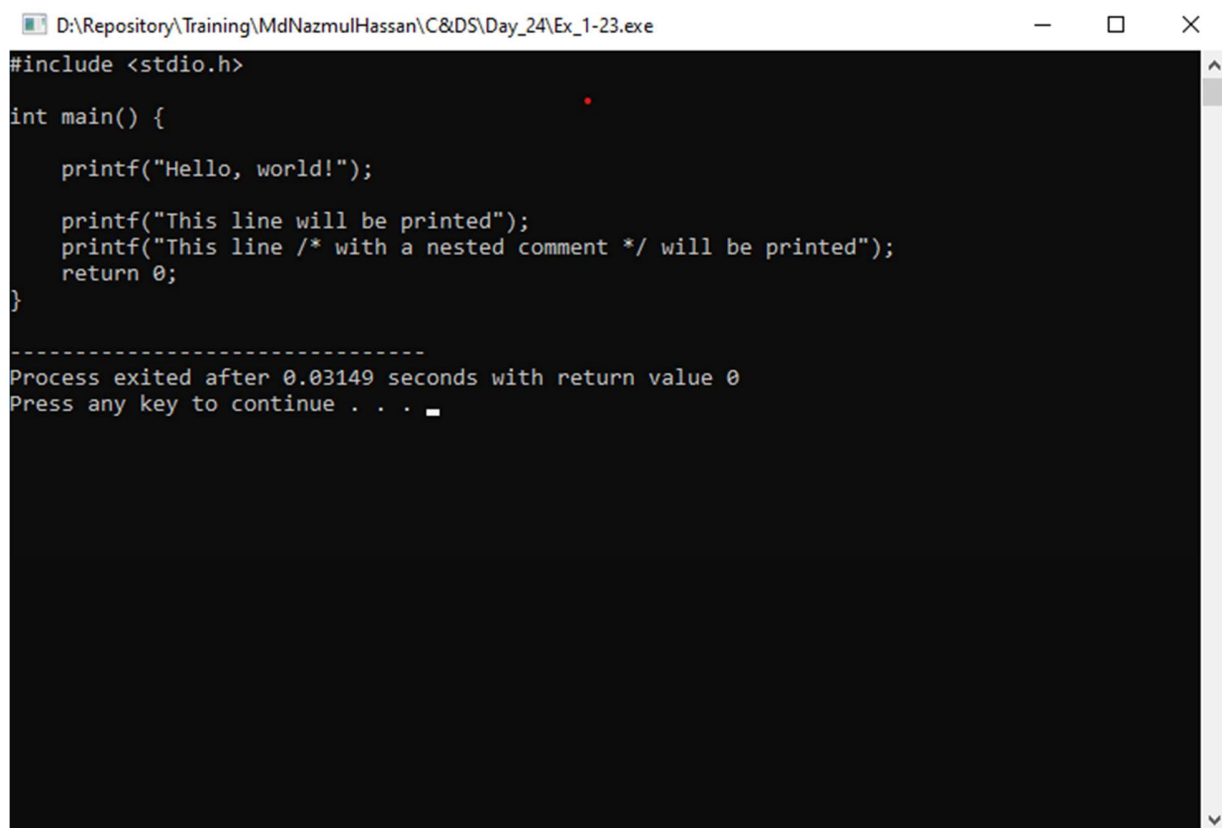
```

→ Concepts Explored:

- **String manipulation:** Manipulating character arrays to remove comments by traversing, comparing, and copying characters.
- **State management:** Using boolean variables to track program state (inside strings, comments) for correct handling.
- **Conditional statements:** Utilizing if-else statements to take actions based on program state and current character.
- **Helper functions:** Implementing functions to identify quoted strings and character constants.
- **Looping:** Iterating over characters using loops for sequential processing.
- **Array manipulation:** Modifying arrays to remove comments while preserving the code.

These concepts are integral to successfully removing comments from C code while preserving string and character constant integrity.

Output:



```
D:\Repository\Training\MdNazmulHassan\C&DS\Day_24\Ex_1-23.exe
#include <stdio.h>

int main() {

    printf("Hello, world!");

    printf("This line will be printed");
    printf("This line /* with a nested comment */ will be printed");
    return 0;
}

-----
Process exited after 0.03149 seconds with return value 0
Press any key to continue . . .
```