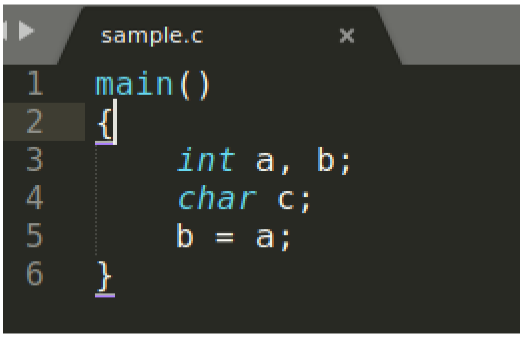
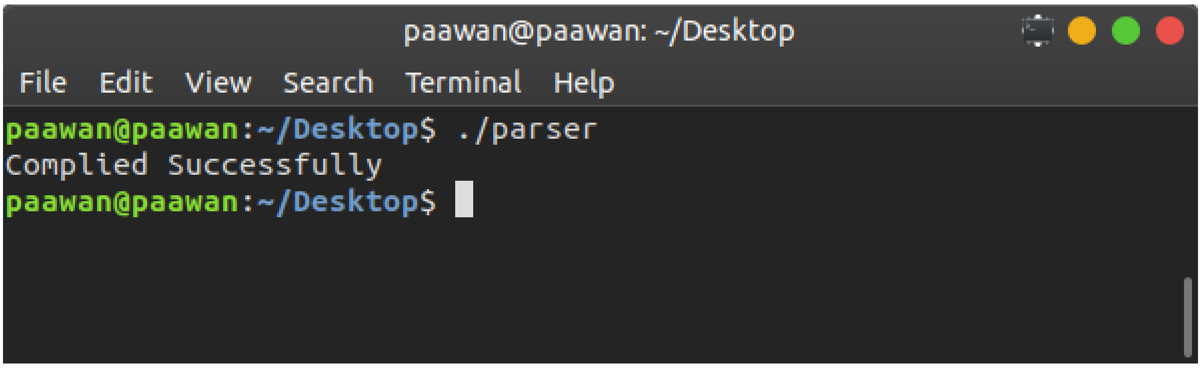
CD Lab 7
Name: Paawan Kohli
Reg No.: 180905416

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

void program();
void declarations();
void data_type();
void identifier_list();
void assign_stat();

struct token curr;
FILE *f1;

void invalid() {
      printf("error");
      exit(0);
}

void program() {
      if (strcmp(curr.lexeme, "ain") == 0) {
            curr = getNextToken(f1);
            if (strcmp(curr.lexeme, "(") == 0) {
                  curr = getNextToken(f1);
                  if (strcmp(curr.lexeme, ")") == 0) {
                        curr = getNextToken(f1);
                        if (strcmp(curr.lexeme, "{") == 0) {
                              curr = getNextToken(f1);
                              declarations();
                              assign_stat();
                              if (strcmp(curr.lexeme, "}") == 0) {
                                    return;
                              } else {
                                    printf("\nMissing } at row:%d and col:%d.\n\n",
                                          curr.row, curr.col);
                                    exit(0);
                              }
                        } else {
                              printf("\nMissing { at row:%d and col:%d.\n\n", curr.row,
                                          curr.col);
                              exit(0);
                        }
                  } else {
                        printf("\nMissing ) at row:%d and col:%d.\n\n", curr.row,
                                    curr.col);
                        exit(0);
                  }
            } else {
                  printf("\nMissing ( at row:%d and col:%d.\n\n", curr.row, curr.col);
                  exit(0);
            }
      } else {
            printf("\nMissing main function\n\n");
            exit(0);
      }
}

void declarations() {
      if (isdtype(curr.lexeme) == 0) {
            return;
      }
      data_type();
      identifier_list();

      if (strcmp(curr.lexeme, ";") == 0) {
            curr = getNextToken(f1);
            declarations();
      }
      else {
            printf("\nMissing ; at row:%d and col:%d.\n\n", curr.row, curr.col);
            exit(0);
      }
}

void data_type() {
      if (strcmp(curr.lexeme, "int") == 0) {
            curr = getNextToken(f1);
            return;
      } else if (strcmp(curr.lexeme, "char") == 0) {
            curr = getNextToken(f1);
            return;
      } else {
            printf("\nMissing data type at row:%d and col:%d.\n\n", curr.row, curr.col);
            exit(0);
      }
}

void identifier_list() {
      if (strcmp(curr.type, "identifier") == 0) {
            curr = getNextToken(f1);

            if (strcmp(curr.lexeme, ",") == 0) {
                  curr = getNextToken(f1);
                  identifier_list();
            } else {
                  return;
            }
      } else {
            printf("\nMissing identifier at row:%d and col:%d.\n\n", curr.row,
                        curr.col);
            exit(0);
      }
}

void assign_stat() {
      if (strcmp(curr.type, "identifier") == 0) {
            curr = getNextToken(f1);
            if (strcmp(curr.lexeme, "=") == 0) {
                  curr = getNextToken(f1);
                  if (strcmp(curr.type, "identifier") == 0) {
                        curr = getNextToken(f1);
                        if (strcmp(curr.lexeme, ";") == 0) {
                              curr = getNextToken(f1);
                              return;
                        }
                  } else if (strcmp(curr.type, "number") == 0) {
                        curr = getNextToken(f1);
                        if (strcmp(curr.lexeme, ";") == 0) {
                              curr = getNextToken(f1);
                              return;
                        } else {
                              printf("\nMissing ; at row:%d and col:%d.\n\n", curr.row,
                                          curr.col);
                              exit(0);
                        }

                  } else {
                        printf("\nMissing identifier at row:%d and col:%d.\n\n",
                                    curr.row, curr.col);
                        exit(0);
                  }
            } else {
                  printf("\nMissing = at row:%d and col:%d.\n\n", curr.row, curr.col);
                  exit(0);
            }
      } else {
            printf("\nMissing identifier at row:%d and col:%d.\n\n", curr.row,
                        curr.col);
            exit(0);
      }
}

int main() {
      FILE *fin, *fout;
      char ca, cb;

      fin = fopen("test.c", "r");

      if (fin == NULL) {
            printf("Cannot open file \n");
            return 0;
      }

      fout = fopen("fileout.c", "w");
      ca = getc(fin);

      while (ca != EOF) {
            if (ca == ' ') {
                  putc(ca, fout);
                  while (ca == ' ') {
                        ca = getc(fin);
                  }
            }

            if (ca == '/') {
                  cb = getc(fin);

                  if (cb == '/') {
                        while (ca != '\n') {
                              ca = getc(fin);
                        }
                  } else if (cb == '*') {
                        do {
                              while (ca != '*') {
                                    ca = getc(fin);
                              }

                              ca = getc(fin);
                        } while (ca != '/');
                  } else {
                        putc(ca, fout);
                        putc(cb, fout);
                  }
            }
            else {
                  putc(ca, fout);
            }

            ca = getc(fin);
      }

      fclose(fin);
      fclose(fout);

      fin = fopen("fileout.c", "r");

      if (fin == NULL) {
            printf("Cannot open file");
```

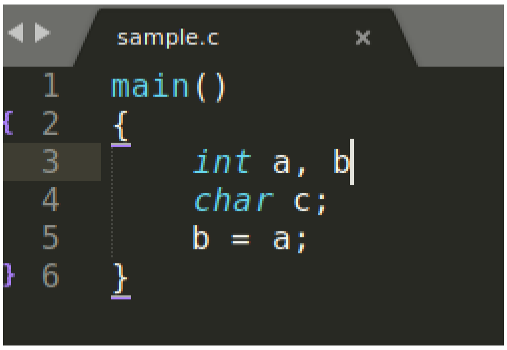
```
        return 0;
    }

    fout = fopen("temp.c", "w");
    ca = getc(fin);

    while (ca != EOF) {
        if (ca == '#') {
            while (ca != '\n') {
                ca = getc(fin);
            }
        }

        ca = getc(fin);

        if (ca != EOF && ca != '#') {
            putc(ca, fout);
        }
    }

    fclose(fin);
    fclose(fout);

    fin = fopen("temp.c", "r");
    fout = fopen("fileout.c", "w");
    ca = getc(fin);

    while (ca != EOF) {
        putc(ca, fout);
        ca = getc(fin);
    }
    fclose(fin);
    fclose(fout);

    remove("temp.c");

    f1 = fopen("fileout.c", "r");

    if (f1 == NULL) {
        printf("Error! File cannot be opened!\n");
        return 0;
    }

    struct token tkn;
    curr = getNextToken(f1);
    program();
    printf("---Compiled Successfully---\n");
    fclose(f1);
}
```



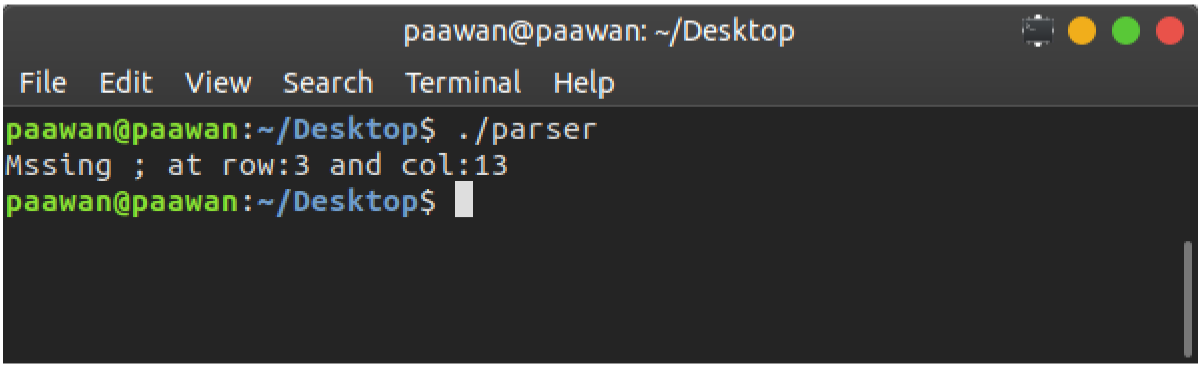
```
main()
{
    int a, b;
    char c;
    b = a;
}
```



```
paawan@paawan:~/Desktop$ ./parser
Complied Successfully
paawan@paawan:~/Desktop$ 
```



```
main()
{
	int a, b
	char c;
	b = a;
}
```



```
paawan@paawan:~/Desktop$ ./parser
Mssing ; at row:3 and col:13
paawan@paawan:~/Desktop$ 
```