```
# $@    Le nom de la cible
# $^    La liste de toutes les dépendances
# $?    La liste des dépendances plus récentes que la cible
# $<    Le nom de la première dépendance
# $*    Le nom du fichier sans suffixe

# user dirs
SRC_DIR         = ./src/
OBJ_DIR         = ./obj/
DEP_DIR         = ./dep/
BIN_DIR         = ./

# bin name
BIN             = question5

# additionnal lib and includes dir
LIB_DIR         = ./
INC_DIR         = ./include/

# compile commands
CC      = g++
CPP     = g++
LD      = g++
FLEX    = flex
BISON   = bison -d

# flags and libs
CPPFLAGS        = -I$(INC_DIR) -g -Wall -pipe
CFLAGS          = -I$(INC_DIR) -g -Wall -pipe
LDFLAGS = -L$(LIB_DIR) -lfl

SRCS_CPP        = $(wildcard $(SRC_DIR)*.cpp)
SRCS_C          = $(wildcard $(SRC_DIR)*.c)
SRCS_L          = $(wildcard $(SRC_DIR)*.l)
SRCS_Y          = $(wildcard $(SRC_DIR)*.y)


#Liste des dépendances .cpp, .c ==> .d
DEPS    = $(SRCS_CPP:$(SRC_DIR)%.cpp=$(DEP_DIR)%.d) $(SRCS_C:$(SRC_DIR)%.c=$(DEP_DIR)%.d)

#Liste des objets : .cpp, .c, .y, .l ==> .o
OBJS    = $(SRCS_CPP:$(SRC_DIR)%.cpp=$(OBJ_DIR)%.o) $(SRCS_C:$(SRC_DIR)%.c=$(OBJ_DIR)%.o) \
          $(SRCS_Y:$(SRC_DIR)%.y=$(OBJ_DIR)%.o) $(SRCS_L:$(SRC_DIR)%.l=$(OBJ_DIR)%.o)

#Rules
all: $(BIN_DIR)/$(BIN)

#To executable
$(BIN_DIR)/$(BIN): $(OBJS)
        $(LD) $+ -o $@ $(LDFLAGS)

#To Objets
$(OBJ_DIR)%.o: $(SRC_DIR)%.c
        $(CC) $(CFLAGS) -o $@ -c $<

$(OBJ_DIR)%.o: $(SRC_DIR)%.cpp
        $(CPP) $(CPPFLAGS) -o $@ -c $<

#To SRC
$(SRC_DIR)%.c : $(SRC_DIR)%.l
        $(FLEX) -o $@ $<
        rm -f $(INC_DIR)parser.h #On efface l'ancien fichier parser.h du dossier "include"
        mv -v $(SRC_DIR)parser.h $(INC_DIR) #Déplacement du nouveau parser.h dans le dossier "include"

$(SRC_DIR)%.c : $(SRC_DIR)%.y
        $(BISON) -o $@ $<

#Gestion des dépendances
$(DEP_DIR)%.d: $(SRC_DIR)%.c
        $(CC) $(CFLAGS) -MM -MD -o $@ $<

$(DEP_DIR)%.d: $(SRC_DIR)%.cpp
        $(CPP) $(CPPFLAGS) -MM -MD -o $@ $<

-include $(DEPS)



.PHONY: clean distclean


run: $(BIN_DIR)/$(BIN)
        $(BIN_DIR)/$(BIN)
```

```
#Rangement des fichiers
store:
	mkdir -p $(INC_DIR)
	mkdir -p $(SRC_DIR)
	mkdir -p $(OBJ_DIR)
	mkdir -p $(DEP_DIR)
	mv -v *.hpp $(INC_DIR)
	mv -v parser.h $(INC_DIR)
	mv -v *.cpp $(SRC_DIR)
	mv -v lexer.l $(SRC_DIR)
	mv -v parser.y $(SRC_DIR)
	mv -v *.o $(OBJ_DIR)
	mv -v *.d $(DEP_DIR)


gdb: $(BIN_DIR)/$(BIN)
	gdb $(BIN_DIR)/$(BIN)

valgrind: $(BIN_DIR)/$(BIN)
	valgrind $(BIN_DIR)/$(BIN)

clean:
	rm -f $(OBJ_DIR)*.o $(SRC_DIR)*~ $(DEP_DIR)*.d *~ $(BIN_DIR)/$(BIN)

distclean: clean
	rm -f $(BIN_DIR)/$(BIN)

tar: clean
	tar -cvzf ../${shell basename `pwd`}.tgz ../${shell basename `pwd`}
```