通用 Makefile 的写法

这个 Makefile 文件首先定义了一些变量（紫色）作为编译参数，接着再定义一些变量（蓝色）表示子目录、源文件、目标文件和头文件依赖，然后根据 Makefile 编写规则写出编译过程，最后的 clean 用于删除编译生成的所有文件。
所有代码如下：

```
CXX=g++
CXXFLAGS=-Wall -O3
DEFINES=
INCLUDES=
THIRD_LIB_DIR=
LIBS=

TARGET=
DIRS=$(shell find . -maxdepth 10 -type d)
SRCS=$(foreach dir,$(DIRS),$(wildcard $(dir)/*.cpp))
OBJS=$(patsubst %.cpp,%.o,$(SRCS))
DEPS=$(patsubst %.cpp,%.d,$(SRCS))

all:$(TARGET)  # all 是伪目标，make 命令会找到 all 并生成对应的$(TARGET)

$%.o:%.cpp
    $(CXX) $(CXXFLAGS) $(INCLUDES) -MMD -MF $(@:.o=.d) -MT $@ -c $< -o $@
    #-MMD 生成.d 依赖文件，-Mf 改变.d 文件的默认命名，-MT 改变目标文件的编译方式。必须这样写否则编译器报错。

-include $(DEPS)

$(TARGET):$(OBJS)
    $(CXX) -o $@ $(OBJS) $(THIRD_LIB_DIR) $(LIBS)

clean:
    rm -rf $(DEPS) $(OBJS) $(TARGET)
```

注意事项：
首先，要清楚 Makefile 的编写规则是：
目标：依赖　　例如：a.o:a.cpp
[Tab 键]命令　　g++ -c a.cpp -o a.o

其次，编译、链接有时候要加上一些选项，常见的有-Wall, -O3, -D, -I, -l, -L，分别定义以下变量来表示这些选项的参数：

```
CXXFLAGS=-Wall -O3 （and others）
DEFINES=-Dxxx -Dxxx  #宏定义
INCLUDES=-Ixx/xx -Ixx/xx  #头文件目录
```

```
LIBS=-lxx  -lxx xx.a xx.a  #-l 是动态库 libxx.so 的写法，xx.a 是静态库的写法
THIRD_LIB_DIR=-Lxx/xx  -Lxx/xx  #第三方库的目录
```

此外，源文件很多的情况下，需用 shell 命令 +Makefile 内嵌函数实现递归查找，找出该项目所有目录下的源文件：

```
DIRS=$(shell find . -maxdepth 10 -type d)  #递归查找当前目录 10 级子目录
SRCS=$(foreach dir,$(DIRS),$(wildcard $(dir)/*.cpp))  #foreach 函数将 DIRS 里的每一项赋给 dir，并调用 wildcard 函数获取 dir 目录下的每一项.cpp 文件
OBJS=$(patsubst %.cpp,%.o,$(SRCS))
DEPS=$(patsubst %.cpp,%.d,$(SRCS))  #patsubst 函数将 SRCS 里每个.cpp 替换成.o（每个源文件对应的目标文件）和.d（头文件依赖，加上这个才能保证修改头文件后会重新编译使更改生效）
```

还要注意一些符号的意义，$()表示变量，$<表示第一个依赖文件，$^表示所有依赖文件（上面未用到），$@表示目标文件，$(@:.o=.d)表示将目标文件.o 替换成依赖文件.d 赋给临时变量。