Question Bank

15.1 What is the basic facility make provide under UNIX?

15.2 How does make help to enhance personal productivity?

15.3 How does make help to enhance project productivity?

15.4 What is the basic file structure of a make file?

15.5 What are the different options available under make? What does -i option indicate under make? Show its use by an example.

15.6 What is the basic inferencing mechanism under make.

15.7 What are interpretations of the macros $@, and $< in make.

15.8 What are some of the standard conventions used by experienced make file users?

15.9 What is a Mastermakefile and what is its use?

15.10 Write a make file for the following

- Makefile
- a.c
- b.c
- c.c
- abc.h
- prog1.c
- prog2.c

a.c is defined as

```
#include <stdio.h>
void
a() {
	printf("function a\n");
}
```

b.c is defined as

```
#include <stdio.h>
void
b() {
        printf("function b\n");
  }
```

c.c is defined as

```
#include <stdio.h>
void
c() {
        printf("function c\n");
}
```

abc.h

```
void a();
void b();
void c();
```

prog1.c is defined as

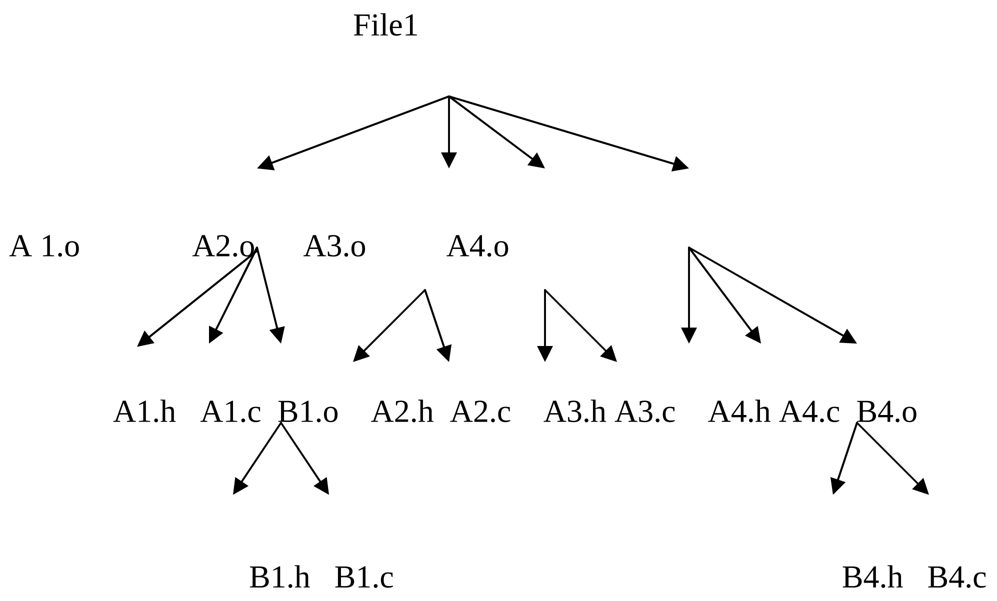
```
#include <stdio.h>
#include <stdlib.h>
#include "abc.h"
int main()
{
        a();
        b();
        c();
        exit(0);
}
```

prog2.c is defined as

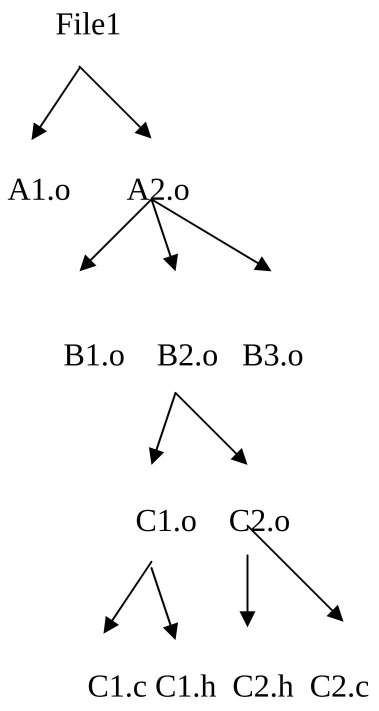
```
#include <stdio.h>
#include "abc.h"
int main()
{
        b();
        a();
        c();
        return(0); /* same as exit(0) */
    }
```

Write a make to compile the above program

15.11 Create makefile for a set of files which have following dependency structure. Use **macros** in the makefile to reduce effort of writing.



15.12 Create makefile for a set of files which have following dependency structure.


File1
A1.o
A2.o
B1.o
B2.o
B3.o
C1.o
C2.o
C1.c
C1.h
C2.h
C2.c