Προγραμματισμός ΙΙ (ECE116)

#5

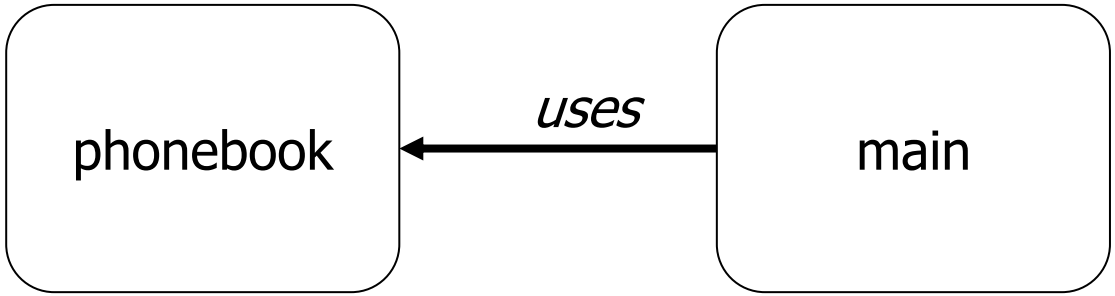
make

Σπύρος Λάλης
lalis@uth.gr

Διαδικασία μεταγλώττισης

- Η διαδικασία μεταγλώττισης ενός σύνθετου συστήματος λογισμικού μπορεί να είναι αρκετά πολύπλοκη
- Πολλά ξεχωριστά αρχεία και βιβλιοθήκες
- Μπορεί εύκολα να γίνουν λάθη
  - να ξεχαστεί μια επιμέρους μεταγλώττιση
- Μπορεί να γίνει «τζάμπα» δουλειά
  - αχρείαστη μεταγλώττιση κώδικα

Παράδειγμα



- `phonebook.h`: τύποι και prototypes συναρτήσεων
- `phonebook.c`: υλοποιήσεις των συναρτήσεων

- `main.c`: κυρίως πρόγραμμα, που χρησιμοποιεί το παραπάνω συστατικό, μέσω των ορισμών που υπάρχουν στο `phonebook.h`

phonebook.h

```
#ifndef __PHONEBOOK_H
#define __PHONEBOOK_H

typedef struct {
  char name[64];  /* name of person */
  char phone[64]; /* phone as string for flexibility */
} entry_t;

typedef struct {
  entry* entries; /* dynamic table holding the entries */
  int size;       /* current size of the table */
} phonebook_t;

extern void phonebook_init(phonebook_t *pb);
extern int phonebook_find(const phonebook_t *pb,
                          const char *name, char *phone);
extern int phonebook_rmv(phonebook_t *pb, const char *name);
extern int phonebook_add(phonebook_t *pb,
                         const char *name, const char *phone);
extern void phonebook_clear(phonebook_t *pb);

#endif
```

phonebook.c

```
#include "phonebook.h"

void phonebook_init(phonebook_t *pb) {
  …
}

int phonebook_find(const phonebook_t *pb,
                   const char *name, char *phone) {
  …
}

int phonebook_rmv(phonebook_t *pb, const char *name) {
  …
}

int phonebook_add(phonebook_t *pb,
                  const char *name, const char *phone) {
  …
}

void phonebook_clear(phonebook_t *pb) {
  …
}
```

main.c

```
#include "phonebook.h"
…

int main(int argc, char *argv[]) {
  …
  entry_t e;
  phonebook_t *pb;

  phonebook_init(&pb);
  …
  phonebook_add(&pb, &e);
  …
  phonebook_find(&pb, e.name, &e);
  …
  phonebook_rmv(&pb, e.name);
  …
  phonebook_add(&pb, &e);
  …
  phonebook_find(&pb, e.name, &e);
  …
  phonebook_clear(&pb);
  …
  return(0);
}
```

phonebook.h

```
...
extern void phonebook_init(phonebook_t *pb);
...
```

main.c

```
...
#include “phonebook.h"

int main (int argc, char *argv[]) {
  ...
  phonebook_t pb;

  phonebook_init(&pb);
  ...
}
```

phonebook.c

```
#include “phonebook.h"
...

void phonebook_init(phonebook_t *pb) {
  ...
}
```

```
> gcc -Wall -c phonebook.c
> gcc -Wall -c main.c
>
> gcc main.o phonebook.o -o all
>
> ./all
```

compile separately

link

execute

```c
#include "phonebook.h"

int main (int argc, char *argv[]) {
 …
  phonebook_t pb;

  phonebook_init(&pb);
  …
}
```

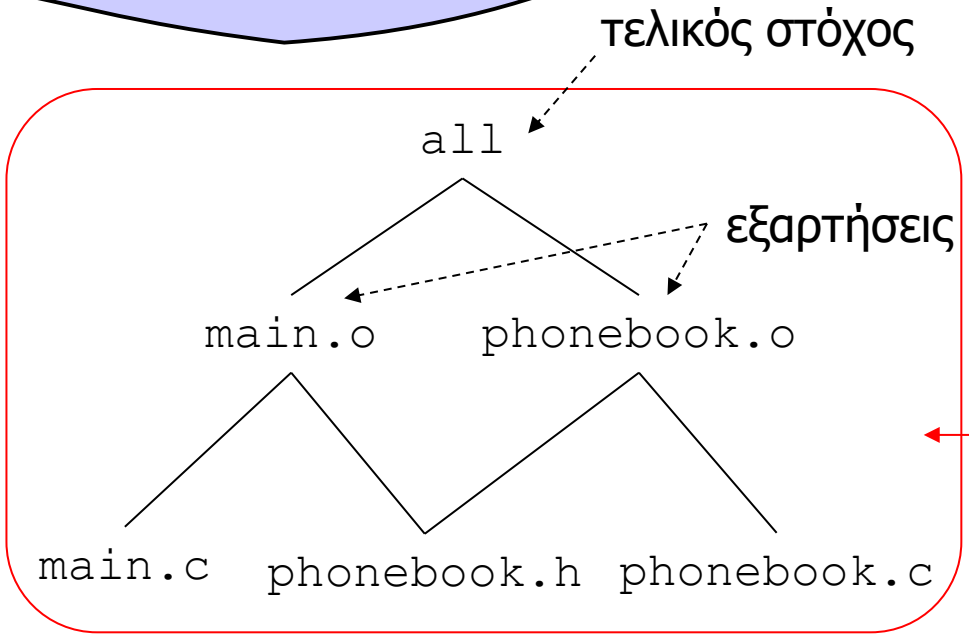
main.c

```c
void phonebook_init(phonebook_t *pb);
```

phonebook.h

```c
#include "phonebook.h"

void phonebook_init(phonebook_t *pb) {
  …
}
```

phonebook.c



πλήρης (δεντρική) δομή εξαρτήσεων

make

- Απαλλάσσει τον προγραμματιστή από το να πρέπει να σκέφτεται και να πληκτρολογεί τις σωστές εντολές, **κάθε φορά** που πρέπει να παραχθεί κώδικας

- Μπορεί να εκτελέσει μια ή περισσότερες λειτουργίες (π.χ. μεταγλώττιση) με **αυτόματο** τρόπο
- Με βάση συγκεκριμένους **κανόνες**

- Οι κανόνες δίνονται σε ένα αρχείο, το **Makefile**
- Περιγράφονται με μια συγκεκριμένη σύνταξη
  - μια γλώσσα ειδικού σκοπού

Βασικά στοιχεία Makefile

- **Κανόνες:** αποτελούνται από το όνομα του **στόχου** (**τι** θέλουμε να δημιουργήσουμε) και τις σχετικές **συνταγές παραγωγής** (**πώς** το δημιουργούμε)
- Οι συνταγές χρησιμοποιούν shell commands

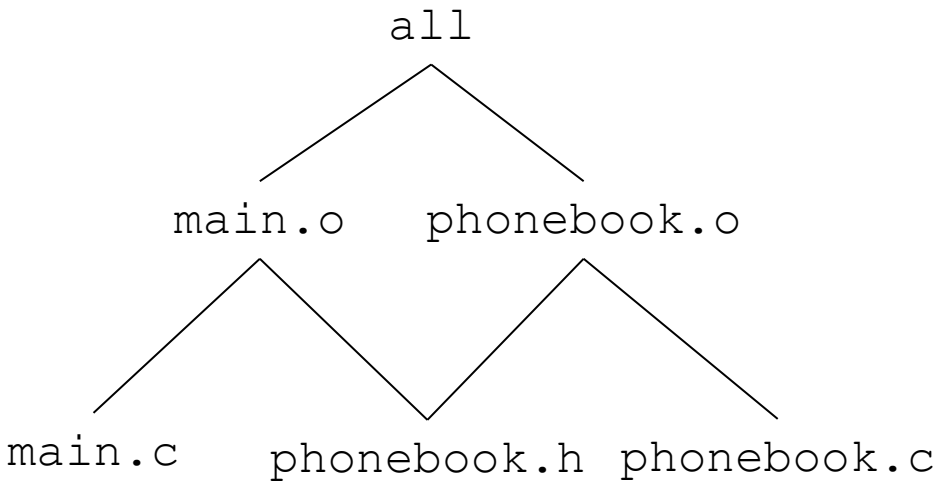
- **Σχόλια:** για τον συνηθισμένο λόγο

- **Μεταβλητές:** χρησιμοποιούνται έτσι ώστε να γίνεται παραμετροποιημένη αναφορά σε ονόματα αρχείων, προγραμμάτων, flags κτλ.

Κανόνες

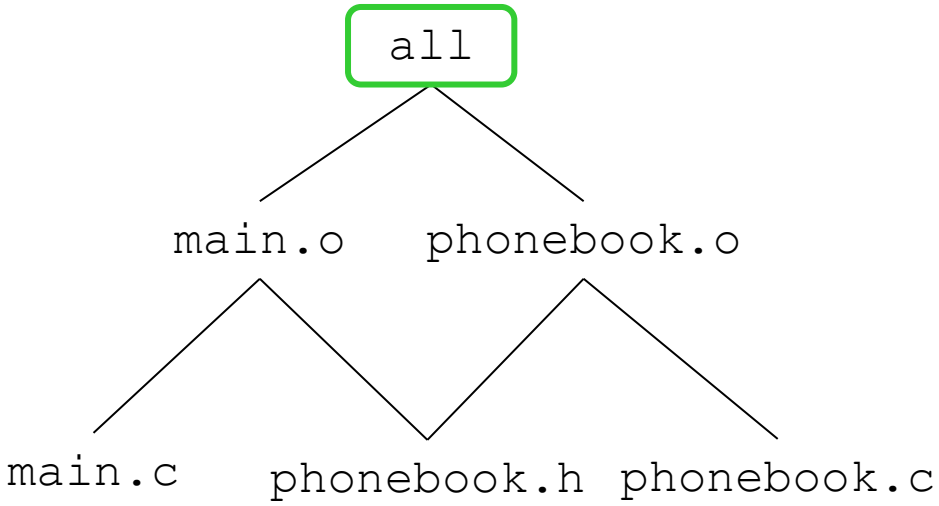
```
<target>: <dependencies>
        <command1>
        <command2>
        …
        <command>
```

- `<target>` (**στόχος**): ένα αρχείο προς κατασκευή ή κάποια ενέργεια προς ολοκλήρωση
- `<dependencies>` (**εξαρτήσεις**): αρχεία ή ενέργειες από τις οποίες εξαρτάται ο στόχος
- `<command>`: **εντολή** που στέλνει το make προς το shell για να δημιουργηθεί ο στόχος (το σύνολο των εντολών για ένα στόχο ονομάζεται και «συνταγή»)
- **Προσοχή:** tab πριν από **κάθε** εντολή



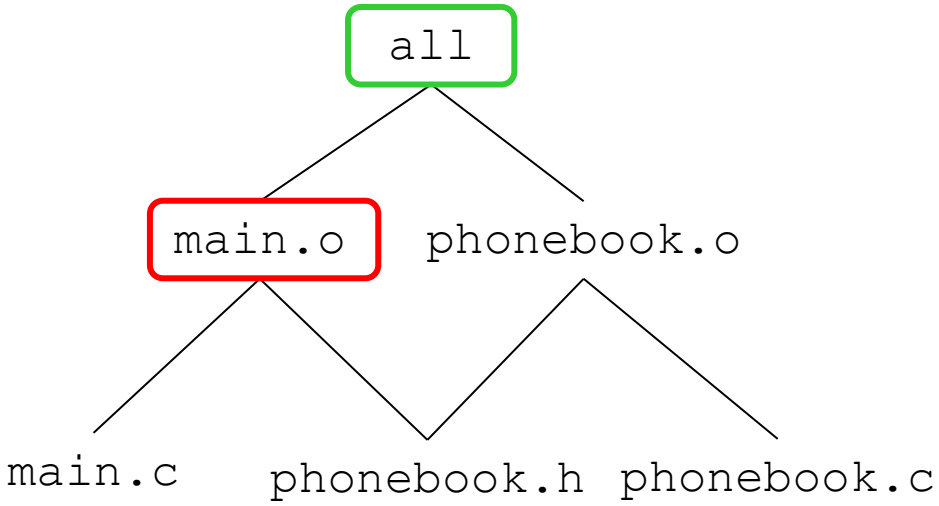
Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



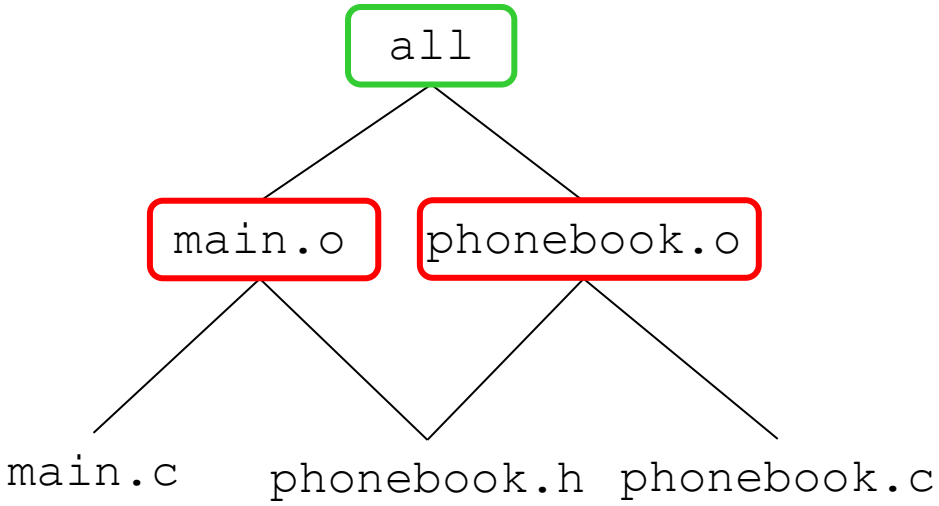
Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



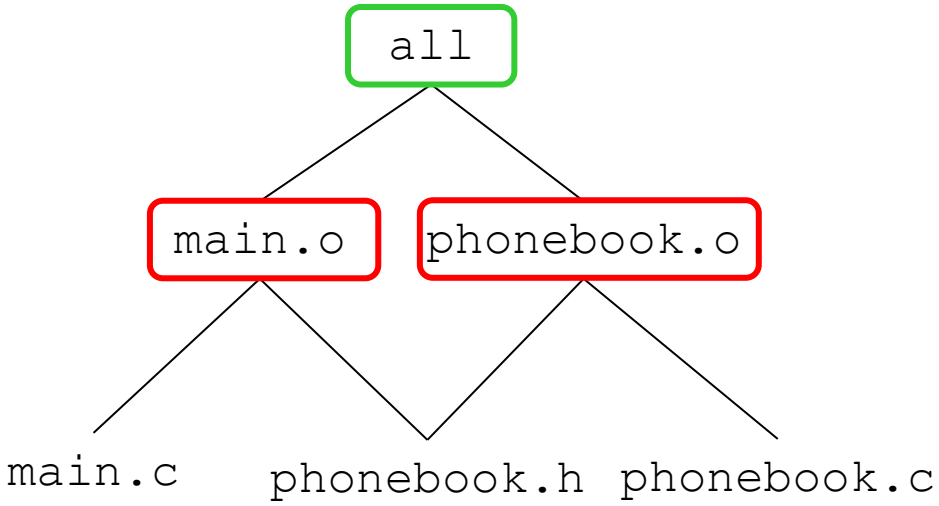
Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



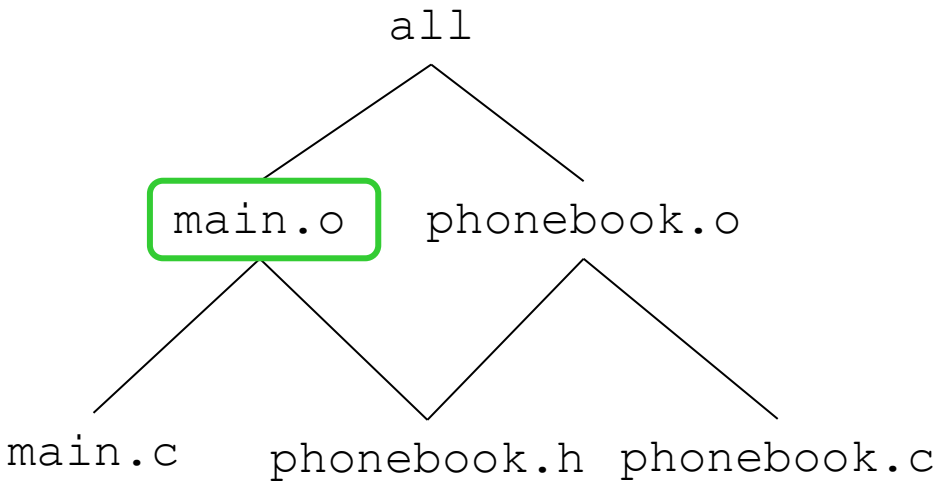
Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



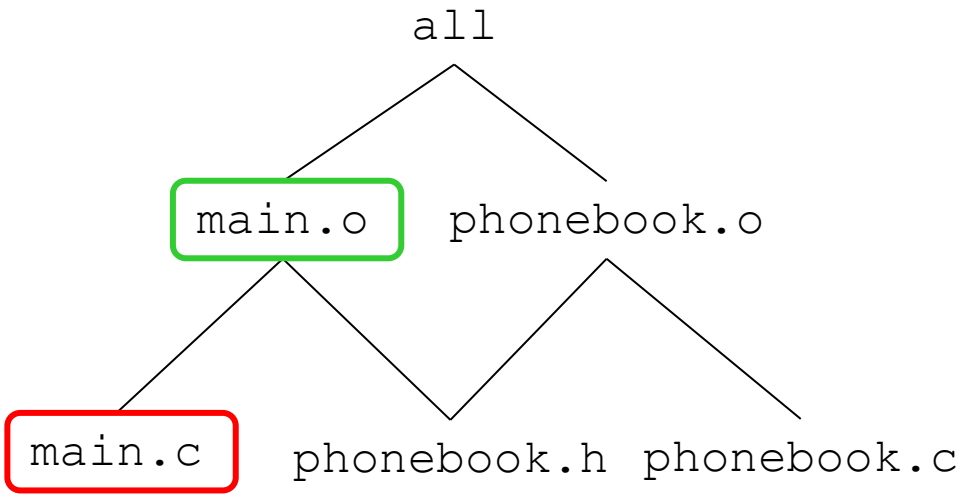
Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



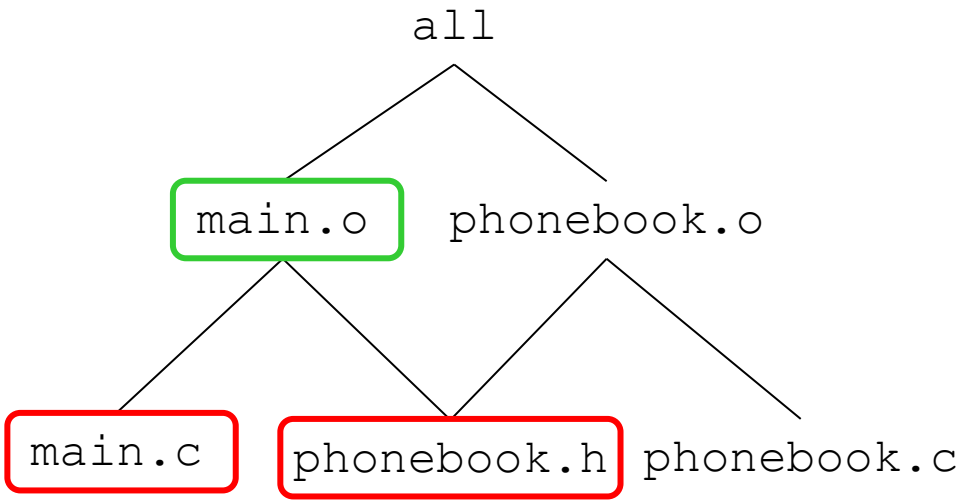
Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



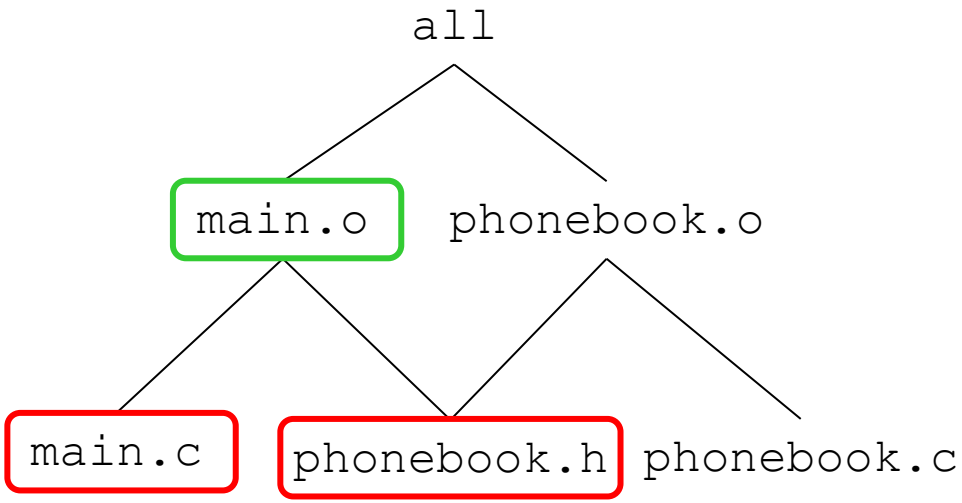
Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



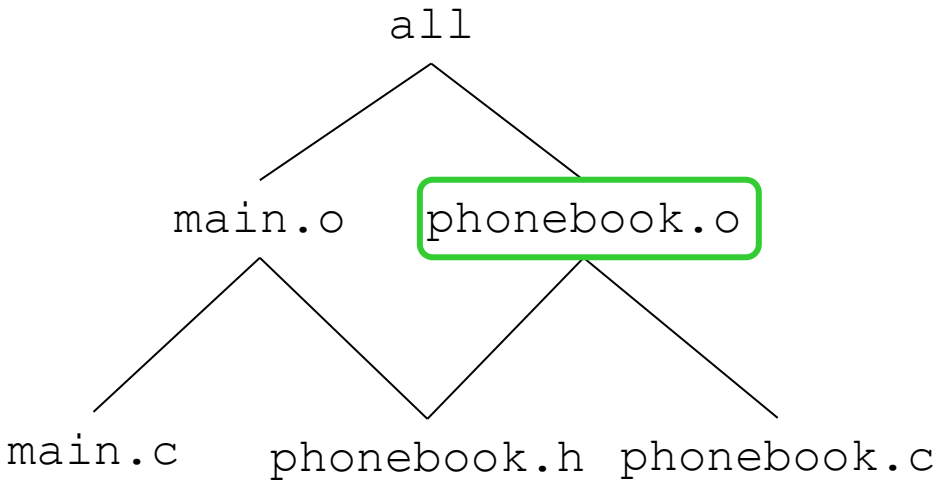
Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



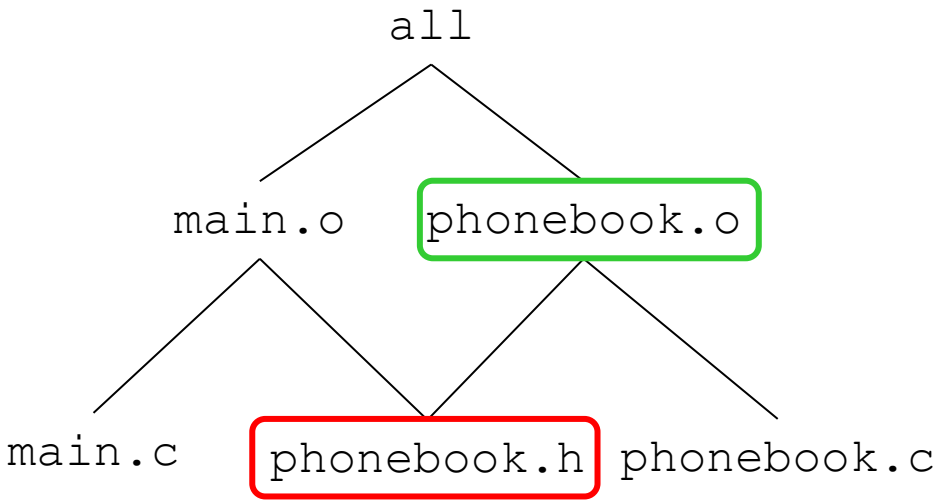
Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



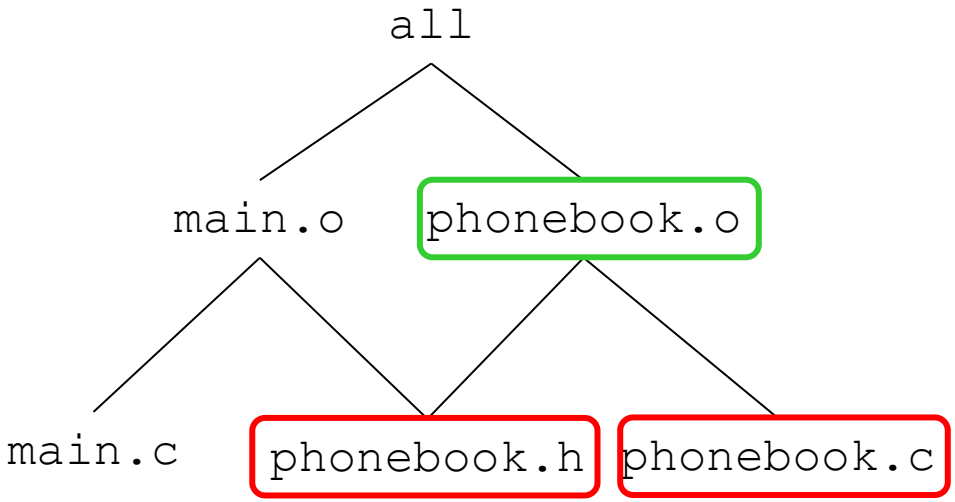
Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



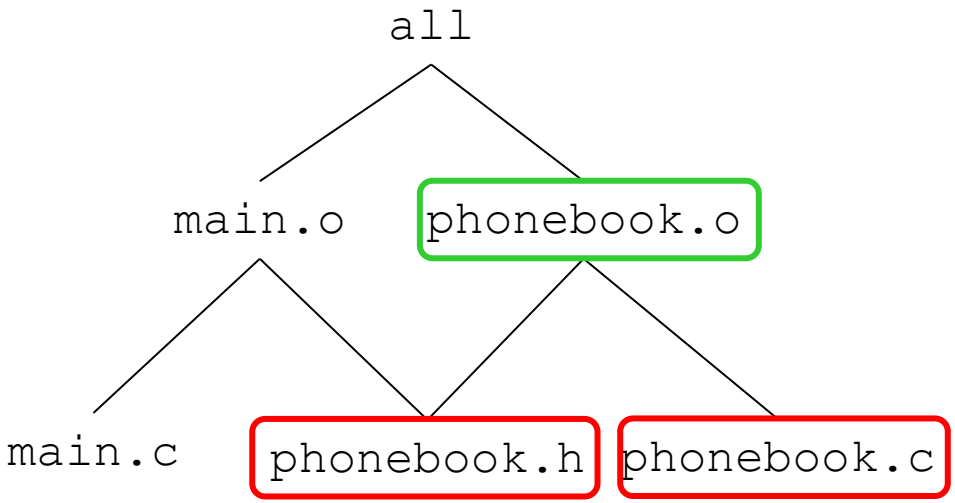
Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```

Εκτέλεση Makefile

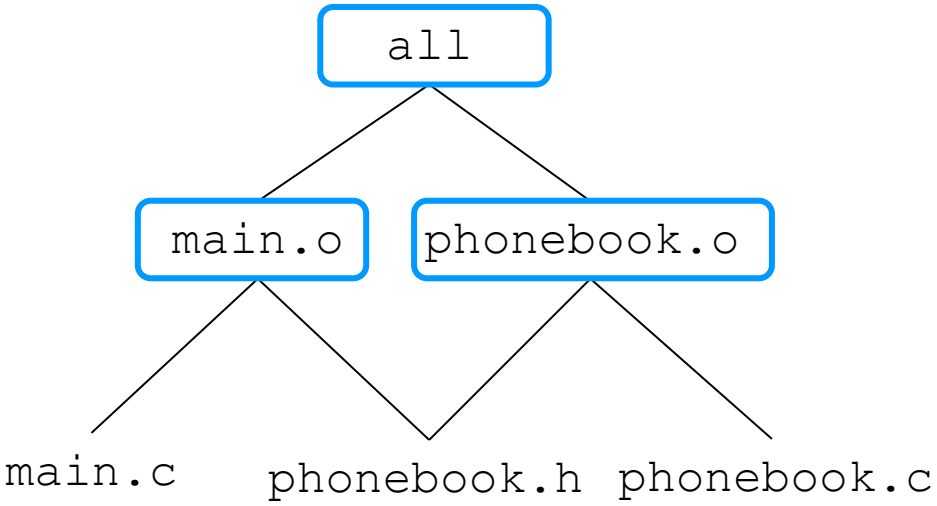
- `make <goal>`
- Το make ψάχνει στο τρέχοντα κατάλογο να βρει ένα αρχείο με όνομα Makefile (ή makefile)
- Ψάχνει στο Makefile να βρει τον στόχο `<goal>`
- Εκτελεί τις εντολές για το συγκεκριμένο στόχο
  - αυτό γίνεται **αναδρομικά** (βλέπε επόμενο)

- `make`
- Όπως παραπάνω, αλλά εκτελεί τις εντολές για τον **πρώτο** στόχο που εμφανίζεται στο Makefile

Αναδρομική & έξυπνη επίτευξη στόχων

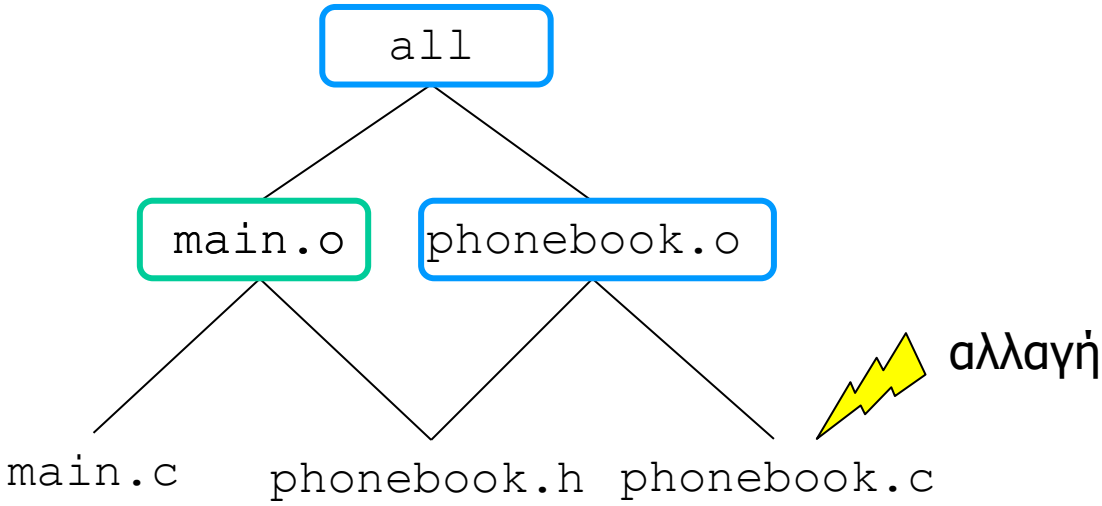
- Αν δεν ικανοποιούνται οι εξαρτήσεις / προϋποθέσεις ενός στόχου, τότε δημιουργούνται **δυναμικά**
- Ακολουθώντας **αναδρομικά** τους αντίστοιχους στόχους

- Αν ένας στόχος ήδη υπάρχει, ελέγχεται το αν έχουν γίνει πρόσφατες αλλαγές στις εξαρτήσεις
  - τους αντίστοιχους προαπαιτούμενους στόχους
- Αν όχι, ο στόχος χρησιμοποιείται **ως έχει**
  - δεν επαναλαμβάνεται χωρίς λόγο η διαδικασία παραγωγής
- Αν ναι, εκτελείται (**ξανά**) η ενέργεια που έχει προσδιοριστεί για την επίτευξη του στόχου



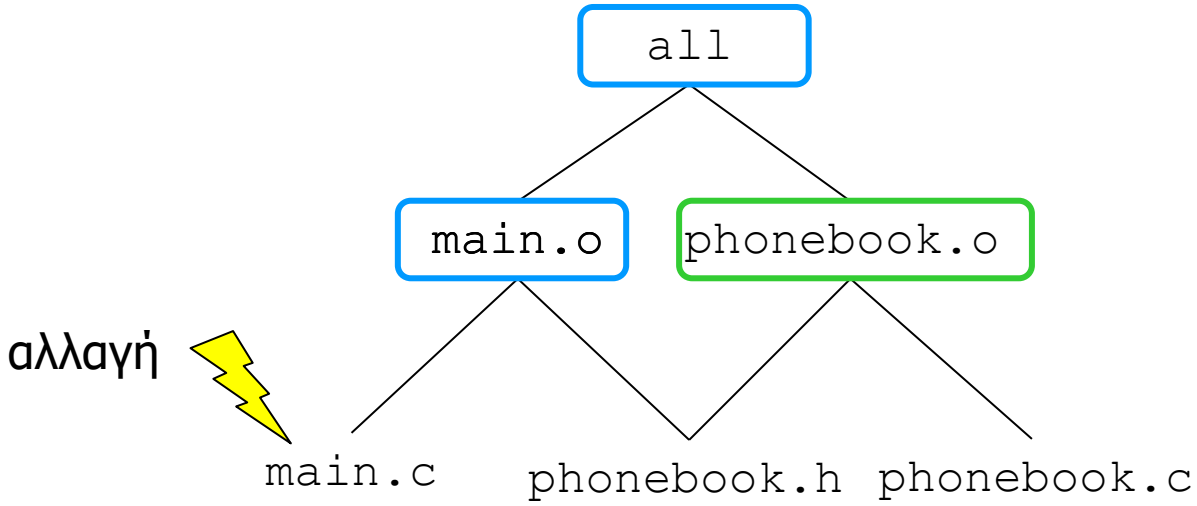
Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



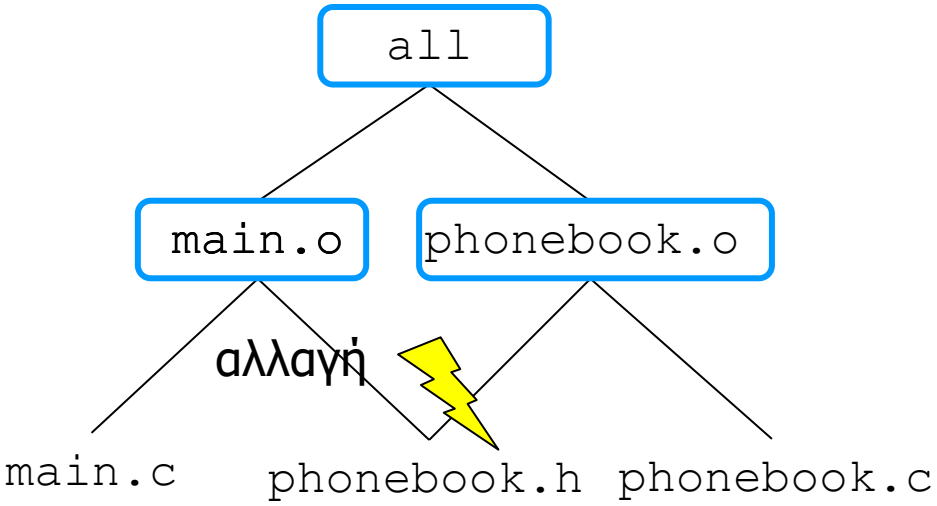
Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



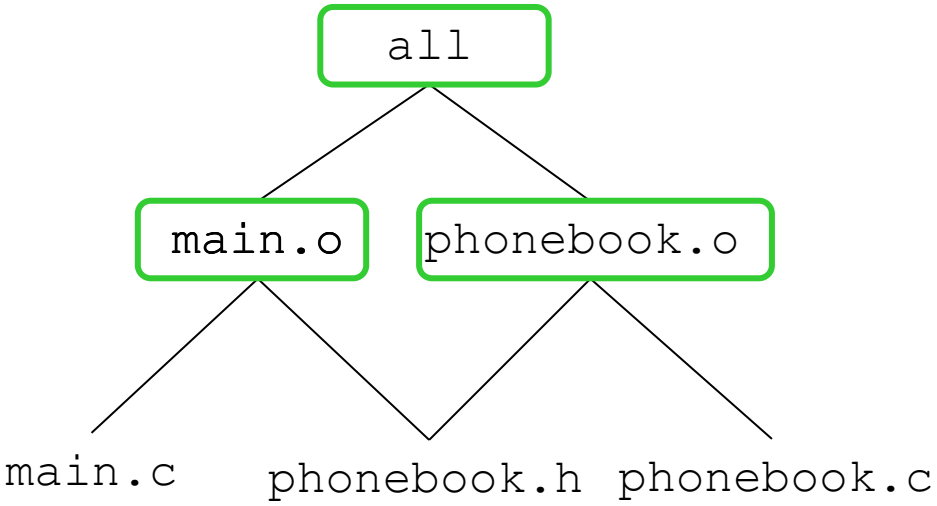
Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```



Makefile

```
all: main.o phonebook.o
	gcc -Wall -g main.o phonebook.o -o all

main.o: main.c phonebook.h
	gcc -Wall -g -c main.c

phonebook.o: phonebook.c phonebook.h
	gcc -Wall -g -c phonebook.c
```

Σειρά εκτέλεσης

```
all: target1
    @echo zero

target1: target2 target3
    @echo one

target2:
    @echo two

target3:
    @echo three
```

output

Σειρά εκτέλεσης

```
all: target1
    @echo zero

target1: target2 target3
    @echo one

target2:
    @echo two

target3:
    @echo three
```

output

Σειρά εκτέλεσης

```
all: target1
    @echo zero

target1: target2 target3
    @echo one

target2:
    @echo two

target3:
    @echo three
```

output

Σειρά εκτέλεσης

```
all: target1
    @echo zero

target1: target2 target3
    @echo one

target2:
    @echo two

target3:
    @echo three
```

output

Σειρά εκτέλεσης

```
all: target1
    @echo zero

target1: target2 target3
    @echo one

target2:
    @echo two

target3:
    @echo three
```

output

Σειρά εκτέλεσης

```
all: target1
    @echo zero

target1: target2 target3
    @echo one

target2:
    @echo two

target3:
    @echo three
```

output

Σειρά εκτέλεσης

```
all: target1
    @echo zero

target1: target2 target3
    @echo one

target2:
    @echo two

target3:
    @echo three
```

output

```
two
```

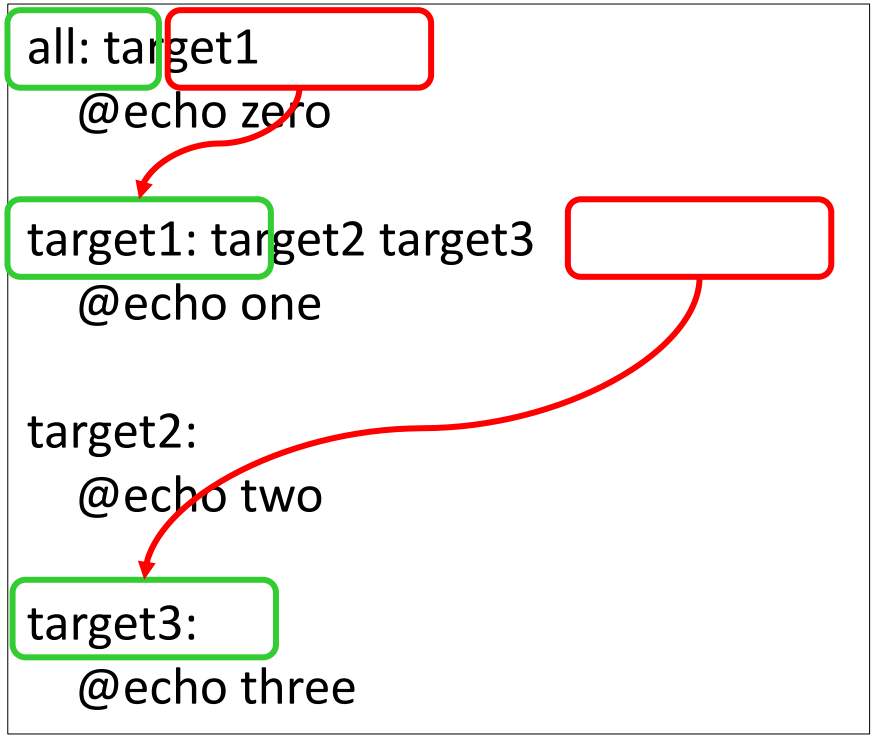
Σειρά εκτέλεσης

```
all: target1
    @echo zero

target1: target2 target3
    @echo one

target2:
    @echo two

target3:
    @echo three
```

output

```
two
```

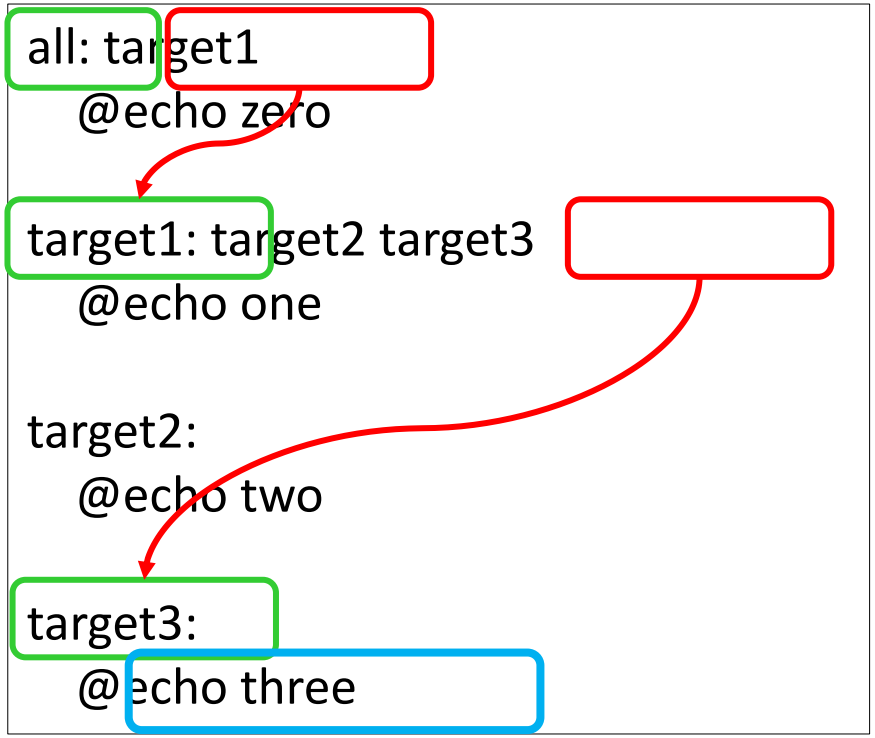
Σειρά εκτέλεσης



output

two

Σειρά εκτέλεσης



output

two
three

Σειρά εκτέλεσης

```
all: target1
    @echo zero

target1: target2 target3
    @echo one

target2:
    @echo two

target3:
    @echo three
```

output

```
two
three
one
```

Σειρά εκτέλεσης

```
all: target1
    @echo zero

target1: target2 target3
    @echo one

target2:
    @echo two

target3:
    @echo three
```

output

```
two
three
one
zero
```

Σειρά εκτέλεσης

```
all: target1
    @echo zero

target1: target2 target3
    @echo one

target2:
    @echo two

target3:
    @echo three
```

output

```
two
three
one
zero
```

output

```
three
two
one
zero
```

εναλλακτική πιθανή εκτέλεση

Μεταβλητές

<name> = <value>

- `<name>` (όνομα): το όνομα της μεταβλητής
- `<value>` (τιμή): ένα ή περισσότερα αρχεία ή ενέργειες από τις οποίες εξαρτάται ο στόχος

- αναφορές στο όνομα μιας μεταβλητής (μέσα σε άλλες εκφράσεις) γίνονται με `$()`

```
CC = gcc
CFLAGS = -Wall -g
OBJ = main.o phonebook.o

all: $(OBJ)
	$(CC) $(OBJ) -o all

main.o: main.c phonebook.h
	$(CC) $(CFLAGS) -c main.c

phonebook.o: phonebook.c phonebook.h
	$(CC) $(CFLAGS) -c phonebook.c
```

Σχόλια

- Ξεκινούν με # και τελειώνουν στο τέλος της γραμμής
- Μπορούν να τοποθετηθούν (σχεδόν) οπουδήποτε

```
CC = gcc
CFLAGS = -Wall -g
OBJ = main.o phonebook.o #all object files

#top-level goal
all: $(OBJ)
	$(CC) $(OBJ) -o all

main.o: main.c phonebook.h
	$(CC) $(CFLAGS) -c main.c

phonebook.o: phonebook.c phonebook.h
	$(CC) $(CFLAGS) -c phonebook.c
```

Patterns και αυτόματες μεταβλητές

- Το σύμβολο `$@` αναπαριστά το όνομα του στόχου
- Το σύμβολο `$<` αναπαριστά το όνομα της πρώτης εξάρτησης για τον τρέχοντα στόχο
- Το σύμβολο `%` μπορεί να χρησιμοποιηθεί στο στόχο ή/και στις εξαρτήσεις για να αναπαραστήσει μέρος του ονόματος ενός αρχείου
  - γίνεται αυτόματο ταίριασμα αναλόγως με την θέση που εμφανίζεται το `%` στο συνολικό όνομα/μοτίβο
- Το σύμβολο `$*` αναπαριστά το κομμάτι του ονόματος που αντιστοιχεί στο `%` (δηλαδή την τιμή που λαμβάνει το `%` αναλόγως το ταίριασμα που έγινε)
- ... και πολλά άλλα ...

εναλλακτική διατύπωση

```
CC = gcc
OBJ = foo.o bar.o

all: $(OBJ)
	$(CC) $(OBJ) -o all

foo.o: foo.c foo.h
	$(CC) -c foo.c

bar.o: bar.c bar.h
	$(CC) -c bar.c
```

```
CC = gcc
OBJ = foo.o bar.o

all: $(OBJ)
	$(CC) $(OBJ) -o $@

foo.o: foo.c foo.h
	$(CC) -c $<

bar.o: bar.c bar.h
	$(CC) -c $<
```

εναλλακτική διατύπωση

```
CC = gcc
OBJ = foo.o bar.o

all: $(OBJ)
	$(CC) $(OBJ) -o $@

%.o: %.c %.h
	$(CC) -c $*.c
```

Στόχοι που δεν είναι αρχεία

```
.PHONY: clean
clean:
        rm *.o
```

- Ένας κανόνας μπορεί οδηγεί σε **οποιαδήποτε πράξη**, όχι απαραίτητα στην μεταγλώττιση αρχείων
- Τότε ο στόχος **δεν** είναι όνομα αρχείου
- Συνήθως χρησιμοποιούμε το χαρακτηρισμό `.PHONY` για να αποφύγουμε «συγκρούσεις» με αρχεία που πιθανώς έχουν ίδιο όνομα με το στόχο
- Τυπική εφαρμογή: κανόνας για το σβήσιμο περιττών αρχείων μετά την ολοκλήρωση της μεταγλώττισης

Στόχοι που δεν είναι αρχεία

```
test%: test%.data myprog
	./myprog < test$*.data

testall: test1 test2 test3
```

- Ένας κανόνας μπορεί να αποτελείται **μόνο** από το στόχο και τις εξαρτήσεις
- Να μην έχει κάποιες εντολές (συνταγή)

Εντοπισμός εξαρτήσεων

- Προϋπόθεση για να φτιαχτεί ένα σωστό Makefile είναι να καταγραφούν σωστά οι **εξαρτήσεις** για την κατασκευή ενός στόχου
  - π.χ., προϋπόθεση για την μεταγλώττιση ενός προγράμματος που χρησιμοποιεί μια βιβλιοθήκη, είναι το αντίστοιχο `.h` αρχείο

- `gcc -MM <file names>`
- Επιστρέφει μια λίστα εξαρτήσεων για τα αρχεία
  - για να συμπεριληφθούν και τα system headers `-M`

Συνηθισμένα λάθη

- `Makefile:4: *** missing separator.`
  - δεν υπάρχει tab στην αρχή εντολής
- `make: *** No rule to make target 'all'.`
  - δεν υπάρχει ο στόχος
- `Makefile:7: warning: overriding commands for target 'test1'`
  - πολλές «συνταγές» για τον ίδιο στόχο
- `Makefile:4: warning: ignoring old commands for target 'test1'`
  - εκτελείται όποια συνταγή είναι τελευταία
  - επιτρέπεται να υπάρχουν πολλοί κανόνες για τον ίδιο στόχο (αλλά τότε πρέπει να έχουν μόνο εξαρτήσεις όχι συνταγές)