

Cognome: _____ Nome: _____ Matricola: _____ Calcolatore: _____

Sistemi Operativi — A.A. 2006-2007, prova pratica del 23 aprile 2007

Compito A

Vietato comunicare con chiunque. Vietato l'uso di rete, cellulari, floppy disk, pen drive e affini. Libri chiusi. Si può usare tutta la documentazione disponibile sul calcolatore. Non spegnere mai il calcolatore. Se hai problemi con il calcolatore rivolgiti subito al docente. Tempo a disposizione: 60 minuti.

Leggere attentamente prima di iniziare

- Scarica dall'url <http://192.168.161.70/compito.tar.gz> il pacchetto dei file che ti servono per il compito e scompattalo all'interno della tua home directory. (suggerimento: “wget <http://192.168.161.70/compito.tar.gz> ; tar xvzf compito.tar.gz”)
- Scrivi nome, cognome, matricola e numero del calcolatore su questo foglio.
- **Scrivi gli stessi dati nel file dati_studente.txt.**
- Prepara un documento di identità a portata di mano.
- **Metti tutti i file che crei durante il compito nella directory ~/compito dentro la home** (nel seguito la tilde “~” significa “home dell'utente”).
- Gli esercizi sono etichettati con 1,2 o 3 asterischi:
 - * = esercizio semplice, valutazione alta, risolvi questi prima degli altri
 - ** = esercizio di media difficoltà
 - *** = esercizio difficile, valutazione bassa, risolvi dopo aver risolto gli altri

Istruzioni per la consegna del compito

Non spegnere il calcolatore e recati dal docente con questo foglio compilato.

Esercizio 1

Il file di testo radb.db.txt contiene un record per ciascun prefisso IP registrato presso un certo registro di Internet. Nel file ciascun record è separato da una linea vuota, i campi sono su linee distinte, tranne il campo “descr” che è su più linee. I campi occupano sempre la stessa posizione. Per svolgere l'esercizio non è necessario conoscere il significato di tutti i campi. Suggerimenti: alcune volte conviene processare tale file con awk usando RS="" (stringa vuota) e FS="\n", ricorda che, in awk, gsub() è un efficace strumento di sostituzione.

1. * Il campo “route” contiene il prefisso. Mostra un comando che selezioni le linee del file del campo “route” con i prefissi /24 che contengano il numero 20 come secondo byte (es. 193.20.10.2) .
Scrivi nel file ~/compito/esercizio1/soluzione1.1.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).
2. ** Dai un comando che mostri una tabella in cui la prima colonna sia il contenuto del campo “member-of” e la seconda il contenuto della prima riga del campo “descr” .
Scrivi nel file ~/compito/esercizio1/soluzione1.2.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).
3. *** [questo punto richiede più tempo degli altri, svolgilo per ultimo] Scrivere uno script che elabori i dati da radb.db.txt e produca su standard output il seguente report:
 - i record sono separati da una riga vuota
 - esiste un record per ciascun maintainer (campo “mnt-by”), tale maintainer è indicato nella prima riga del record
 - le restanti righe del record contengono tutti i prefissi i cui record sono mantenuti da quel maintainer
 - i record devono essere ordinati in modo che i maintainer con maggior numero di “route” siano in cima.

Il nome del tuo script deve essere ~/compito/esercizio1/script.sh

Esercizio 2

Il programma "fib" contenuto in prj1 è composto da più file con estensione ".c" e ".h". Tale programma crea una list contenente i primi n numeri di fibonacci elevati a 1.5 e ne stampa il contenuto (n è passato come parametro). La directory prj2 contiene una copia di prj1 che devi modificare per rispondere alle seguenti domande.

1. * Usa il comando "gcc *.c" per una prima compilazione. Correggi gli errori di preprocessing, di compilazione e di link che incontri. Mostra tutti gli errori e spiegali man mano che li correggi. Mostra quindi l'output di compilazione ed esecuzione del programma corretto. Nota che per eseguire il programma devi fornire un parametro, usa, ad esempio, "6".

Per correggere gli errori modifica la copia in ~/compito/esercizio2/prj2 e lascia intatto ~/compito/esercizio2/prj1.

Scrivi in ~/compito/esercizio2/soluzione2.1.txt la soluzione di questo esercizio (fai copia-e-incolla dal terminale per gli errori e l'output).

2. ** Crea un Makefile con i seguenti target **badando a mettere le dipendenze opportune**

- fib: crea l'eseguibile "fib" linkato dinamicamente con simboli di debug
- clean: pulisce il progetto cancellando i file inutili (*.o, *.~)
- delete: come clean ma cancella anche i target
- test: dipende da fib, ed esegue fib con argomento 15

Il Makefile si deve chiamare ~/compito/esercizio2/prj2/Makefile

3. * Crea una patch tra prj1 e prj2 e prova la su una copia "prj1 copy" che crei tu.

Metti nel file ~/compito/esercizio2/soluzione2.3.txt i comandi per creare e per applicare la patch (fai copia-e-incolla dal terminale). Metti la patch nel file ~/compito/esercizio2/soluzione2.3_patch.txt.

4. ** Esegui il seguente esperimento: considera l'output del comando "**objdump -t init_list.o**". Ora metti "**static**" davanti a "**long int fib(long int f1)**" ed esegui ancora lo stesso comando. Mostra le differenze. Quale è la semantica di questo uso di "**static**"?

Metti nel file ~/compito/esercizio2/soluzione2.4.txt i comandi per portare a termine l'esperimento (fai copia-e-incolla dal terminale) e la risposta alla domanda.

5. ** Considera una esecuzione di fib con parametro 15. Mostra il contenuto dello stack all'inizio della 2000-esima esecuzione della funzione fib().

Scrivi nel file ~/compito/esercizio2/soluzione2.5.txt la sessione di debug (fai copia-e-incolla dal terminale).

Compito B

Vietato comunicare con chiunque. Vietato l'uso di rete, cellulari, floppy disk, pen drive e affini. Libri chiusi. Si può usare tutta la documentazione disponibile sul calcolatore. Non spegnere mai il calcolatore. Se hai problemi con il calcolatore rivolgiti subito al docente. Tempo a disposizione: 60 minuti.

Leggere attentamente prima di iniziare

- Scarica dall'url <http://192.168.161.70/compito.tar.gz> il pacchetto dei file che ti servono per il compito e scompattalo all'interno della tua home directory. (suggerimento: “wget <http://192.168.161.70/compito.tar.gz> ; tar xvzf compito.tar.gz”)
- Scrivi nome, cognome, matricola e numero del calcolatore su questo foglio.
- **Scrivi gli stessi dati nel file dati_studente.txt.**
- Prepara un documento di identità a portata di mano.
- **Metti tutti i file che crei durante il compito nella directory ~/compito dentro la home** (nel seguito la tilde “~” significa “home dell'utente”).
- Gli esercizi sono etichettati con 1,2 o 3 asterischi:
 - * = esercizio semplice, valutazione alta, risolvi questi prima degli altri
 - ** = esercizio di media difficoltà
 - *** = esercizio difficile, valutazione bassa, risolvi dopo aver risolto gli altri

Istruzioni per la consegna del compito

Non spegnere il calcolatore e recati dal docente con questo foglio compilato.

Esercizio 1

Il file di testo radb.db.txt contiene un record per ciascun prefisso IP registrato presso un certo registro di Internet. Nel file ciascun record è separato da una linea vuota, i campi sono su linee distinte, tranne il campo “descr” che è su più linee. I campi occupano sempre la stessa posizione. Per svolgere l'esercizio non è necessario conoscere il significato di tutti i campi. Suggerimenti: alcune volte conviene processare tale file con awk usando RS="" (stringa vuota) e FS="\n", ricorda che, in awk, gsub() e' un efficace strumento di sostituzione.

1. * Il campo “route” contiene il prefisso. Mostra un comando che selezioni le linee del file del campo “route” con i prefissi /24 che contengano il numero 20 come terzo byte (es. 193.10.20.2) .
Scrivi nel file ~/compito/esercizio1/soluzione1.1.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).
2. ** Dai un comando che mostri una tabella in cui la prima colonna sia il contenuto del campo “mnt-by” e la seconda sia il contenuto della prima riga del campo “descr”.
Scrivi nel file ~/compito/esercizio1/soluzione1.2.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).
3. *** [questo punto richiede più tempo degli altri, svolgilo per ultimo] Scrivere uno script che elabori i dati da radb.db.txt e produca su standard output il seguente report:
 - i record sono separati da una riga vuota
 - esiste un record per ciascun maintainer (campo “mnt-by”), tale maintainer è indicato nella prima riga del record
 - le restanti righe del record contengono tutti i prefissi i cui record sono mantenuti da quel maintainer
 - i record devono essere ordinati in modo che i maintainer con maggior numero di “route” siano in cima.

Il nome del tuo script deve essere ~/compito/esercizio1/script.sh

Esercizio 2

Il programma "fib" contenuto in prj1 è composto da più file con estensione ".c" e ".h". Tale programma crea una list contenente i primi n numeri di fibonacci elevati a 1.5 e ne stampa il contenuto (n e' passato come parametro). La directory prj2 contiene una copia di prj1 che devi modificare per rispondere alle seguenti domande.

1. * Usa il comando "gcc *.c" per una prima compilazione. Correggi gli errori di preprocessing, di compilazione e di link che incontri. Mostra tutti gli errori e spiegali man mano che li correggi. Mostra quindi l'output di compilazione ed esecuzione del programma corretto. Nota che per eseguire il programma devi fornire un parametro, usa, ad esempio, "6".

Per correggere gli errori modifica la copia in ~/compito/esercizio2/prj2 e lascia intatto ~/compito/esercizio2/prj1.

Scrivi in ~/compito/esercizio2/soluzione2.1.txt la soluzione di questo esercizio (fai copia-e-incolla dal terminale per gli errori e l'output).

2. ** Crea un Makefile con i seguenti target **badando a mettere le dipendenze opportune**

- fib: crea l'eseguibile "fib" linkato dinamicamente con simboli di debug
- clean: pulisce il progetto cancellando i file inutili (*.o, *.~)
- delete: come clean ma cancella anche i target
- test: dipende da fib, ed esegue fib con argomento 15

Il Makefile si deve chiamare ~/compito/esercizio2/prj2/Makefile

3. * Crea una patch tra prj1 e prj2 e prova la su una copia "prj1copy" che crei tu.

Metti nel file ~/compito/esercizio2/soluzione2.3.txt i comandi per creare e per applicare la patch (fai copia-e-incolla dal terminale). Metti la patch nel file ~/compito/esercizio2/soluzione2.3_patch.txt.

4. ** Esegui il seguente esperimento: considera l'output del comando "**objdump -t init_list.o**". Ora metti "**static**" davanti a "**long int fib(long int f1)**" ed esegui ancora lo stesso comando. Mostra le differenze. Quale e' la semantica di questo uso di "**static**"?

Metti nel file ~/compito/esercizio2/soluzione2.4.txt i comandi per portare a termine l'esperimento (fai copia-e-incolla dal terminale) e la risposta alla domanda.

5. ** Considera una esecuzione di fib con parametro 15. Mostra il contenuto dello stack all'inizio della 2000-esima esecuzione della funzione fib().

Scrivi nel file ~/compito/esercizio2/soluzione2.5.txt la sessione di debug (fai copia-e-incolla dal terminale).

Compito C

Vietato comunicare con chiunque. Vietato l'uso di rete, cellulari, floppy disk, pen drive e affini. Libri chiusi. Si può usare tutta la documentazione disponibile sul calcolatore. Non spegnere mai il calcolatore. Se hai problemi con il calcolatore rivolgiti subito al docente. Tempo a disposizione: 60 minuti.

Leggere attentamente prima di iniziare

- Scarica dall'url <http://192.168.161.70/compito.tar.gz> il pacchetto dei file che ti servono per il compito e scompattalo all'interno della tua home directory. (suggerimento: “wget <http://192.168.161.70/compito.tar.gz> ; tar xvzf compito.tar.gz”)
- Scrivi nome, cognome, matricola e numero del calcolatore su questo foglio.
- **Scrivi gli stessi dati nel file dati_studente.txt.**
- Prepara un documento di identità a portata di mano.
- **Metti tutti i file che crei durante il compito nella directory ~/compito dentro la home** (nel seguito la tilde “~” significa “home dell'utente”).
- Gli esercizi sono etichettati con 1,2 o 3 asterischi:
 - * = esercizio semplice, valutazione alta, risolvi questi prima degli altri
 - ** = esercizio di media difficoltà
 - *** = esercizio difficile, valutazione bassa, risolvi dopo aver risolto gli altri

Istruzioni per la consegna del compito

Non spegnere il calcolatore e recati dal docente con questo foglio compilato.

Esercizio 1

Il file di testo radb.db.txt contiene un record per ciascun prefisso IP registrato presso un certo registro di Internet. Nel file ciascun record è separato da una linea vuota, i campi sono su linee distinte, tranne il campo “descr” che è su più linee. I campi occupano sempre la stessa posizione. Per svolgere l'esercizio non è necessario conoscere il significato di tutti i campi. Suggerimenti: alcune volte conviene processare tale file con awk usando RS="" (stringa vuota) e FS="\n", ricorda che, in awk, gsub() e' un efficace strumento di sostituzione.

1. * Il campo “route” contiene il prefisso. Mostra un comando che selezioni le linee del file del campo “route” con i prefissi /24 che contengano il numero 30 come secondo byte (es. 193.**30**.20.2) .
Scrivi nel file ~/compito/esercizio1/soluzione1.1.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).
2. ** Dai un comando che mostri una tabella in cui la prima colonna sia il contenuto del campo “member-of” e la seconda il contenuto del campo “changed” .
Scrivi nel file ~/compito/esercizio1/soluzione1.2.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).
3. *** [questo punto richiede più tempo degli altri, svolgilo per ultimo] Scrivere uno script che elabori i dati da radb.db.txt e produca su standard output il seguente report:
 - i record sono separati da una riga vuota
 - esiste un record per ciascun maintainer (campo “mnt-by”), tale maintainer è indicato nella prima riga del record
 - le restanti righe del record contengono tutti i prefissi i cui record sono mantenuti da quel maintainer
 - i record devono essere ordinati in modo che i maintainer con maggior numero di “route” siano in cima.

Il nome del tuo script deve essere ~/compito/esercizio1/script.sh

Esercizio 2

Il programma "fib" contenuto in prj1 è composto da più file con estensione ".c" e ".h". Tale programma crea una list contenente i primi n numeri di fibonacci elevati a 1.5 e ne stampa il contenuto (n e' passato come parametro). La directory prj2 contiene una copia di prj1 che devi modificare per rispondere alle seguenti domande.

1. * Usa il comando "gcc *.c" per una prima compilazione. Correggi gli errori di preprocessing, di compilazione e di link che incontri. Mostra tutti gli errori e spiegali man mano che li correggi. Mostra quindi l'output di compilazione ed esecuzione del programma corretto. Nota che per eseguire il programma devi fornire un parametro, usa, ad esempio, "6".

Per correggere gli errori modifica la copia in ~/compito/esercizio2/prj2 e lascia intatto ~/compito/esercizio2/prj1.

Scrivi in ~/compito/esercizio2/soluzione2.1.txt la soluzione di questo esercizio (fai copia-e-incolla dal terminale per gli errori e l'output).

2. ** Crea un Makefile con i seguenti target **badando a mettere le dipendenze opportune**

- fib: crea l'eseguibile "fib" linkato dinamicamente con simboli di debug
- clean: pulisce il progetto cancellando i file inutili (*.o, *.~)
- delete: come clean ma cancella anche i target
- test: dipende da fib, ed esegue fib con argomento 15

Il Makefile si deve chiamare ~/compito/esercizio2/prj2/Makefile

3. * Crea una patch tra prj1 e prj2 e prova la su una copia "prj1copy" che crei tu.

Metti nel file ~/compito/esercizio2/soluzione2.3.txt i comandi per creare e per applicare la patch (fai copia-e-incolla dal terminale). Metti la patch nel file ~/compito/esercizio2/soluzione2.3_patch.txt.

4. ** Esegui il seguente esperimento: considera l'output del comando "**objdump -t init_list.o**". Ora metti "**static**" davanti a "**long int fib(long int f1)**" ed esegui ancora lo stesso comando. Mostra le differenze. Quale e' la semantica di questo uso di "**static**"?

Metti nel file ~/compito/esercizio2/soluzione2.4.txt i comandi per portare a termine l'esperimento (fai copia-e-incolla dal terminale) e la risposta alla domanda.

5. ** Considera una esecuzione di fib con parametro 15. Mostra il contenuto dello stack all'inizio della 2000-esima esecuzione della funzione fib().

Scrivi nel file ~/compito/esercizio2/soluzione2.5.txt la sessione di debug (fai copia-e-incolla dal terminale).

Compito D

Vietato comunicare con chiunque. Vietato l'uso di rete, cellulari, floppy disk, pen drive e affini. Libri chiusi. Si può usare tutta la documentazione disponibile sul calcolatore. Non spegnere mai il calcolatore. Se hai problemi con il calcolatore rivolgiti subito al docente. Tempo a disposizione: 60 minuti.

Leggere attentamente prima di iniziare

- Scarica dall'url <http://192.168.161.70/compito.tar.gz> il pacchetto dei file che ti servono per il compito e scompattalo all'interno della tua home directory. (suggerimento: “wget <http://192.168.161.70/compito.tar.gz> ; tar xvzf compito.tar.gz”)
- Scrivi nome, cognome, matricola e numero del calcolatore su questo foglio.
- **Scrivi gli stessi dati nel file dati_studente.txt.**
- Prepara un documento di identità a portata di mano.
- **Metti tutti i file che crei durante il compito nella directory ~/compito dentro la home** (nel seguito la tilde “~” significa “home dell'utente”).
- Gli esercizi sono etichettati con 1,2 o 3 asterischi:
 - * = esercizio semplice, valutazione alta, risolvi questi prima degli altri
 - ** = esercizio di media difficoltà
 - *** = esercizio difficile, valutazione bassa, risolvi dopo aver risolto gli altri

Istruzioni per la consegna del compito

Non spegnere il calcolatore e recati dal docente con questo foglio compilato.

Esercizio 1

Il file di testo radb.db.txt contiene un record per ciascun prefisso IP registrato presso un certo registro di Internet. Nel file ciascun record è separato da una linea vuota, i campi sono su linee distinte, tranne il campo “descr” che è su più linee. I campi occupano sempre la stessa posizione. Per svolgere l'esercizio non è necessario conoscere il significato di tutti i campi. Suggerimenti: alcune volte conviene processare tale file con awk usando RS="" (stringa vuota) e FS="\n", ricorda che, in awk, gsub() è un efficace strumento di sostituzione.

1. * Il campo “route” contiene il prefisso. Mostra un comando che selezioni le linee del file del campo “route” con i prefissi /24 che contengano il numero 10 come terzo byte (es. 193.20.10.2) .
Scrivi nel file ~/compito/esercizio1/soluzione1.1.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).
2. ** Dai un comando che mostri una tabella in cui la prima colonna sia il contenuto del campo “changed” e la seconda sia il contenuto del campo “mnt-by”.
Scrivi nel file ~/compito/esercizio1/soluzione1.2.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).
3. *** [questo punto richiede più tempo degli altri, svolgilo per ultimo] Scrivere uno script che elabori i dati da radb.db.txt e produca su standard output il seguente report:
 - i record sono separati da una riga vuota
 - esiste un record per ciascun maintainer (campo “mnt-by”), tale maintainer è indicato nella prima riga del record
 - le restanti righe del record contengono tutti i prefissi i cui record sono mantenuti da quel maintainer
 - i record devono essere ordinati in modo che i maintainer con maggior numero di “route” siano in cima.

Il nome del tuo script deve essere ~/compito/esercizio1/script.sh

Esercizio 2

Il programma "fib" contenuto in prj1 è composto da più file con estensione ".c" e ".h". Tale programma crea una list contenente i primi n numeri di fibonacci elevati a 1.5 e ne stampa il contenuto (n e' passato come parametro). La directory prj2 contiene una copia di prj1 che devi modificare per rispondere alle seguenti domande.

1. * Usa il comando "gcc *.c" per una prima compilazione. Correggi gli errori di preprocessing, di compilazione e di link che incontri. Mostra tutti gli errori e spiegali man mano che li correggi. Mostra quindi l'output di compilazione ed esecuzione del programma corretto. Nota che per eseguire il programma devi fornire un parametro, usa, ad esempio, "6".

Per correggere gli errori modifica la copia in ~/compito/esercizio2/prj2 e lascia intatto ~/compito/esercizio2/prj1.

Scrivi in ~/compito/esercizio2/soluzione2.1.txt la soluzione di questo esercizio (fai copia-e-incolla dal terminale per gli errori e l'output).

2. ** Crea un Makefile con i seguenti target **badando a mettere le dipendenze opportune**

- fib: crea l'eseguibile "fib" linkato dinamicamente con simboli di debug
- clean: pulisce il progetto cancellando i file inutili (*.o, *.~)
- delete: come clean ma cancella anche i target
- test: dipende da fib, ed esegue fib con argomento 15

Il Makefile si deve chiamare ~/compito/esercizio2/prj2/Makefile

3. * Crea una patch tra prj1 e prj2 e prova la su una copia "prj1copy" che crei tu.

Metti nel file ~/compito/esercizio2/soluzione2.3.txt i comandi per creare e per applicare la patch (fai copia-e-incolla dal terminale). Metti la patch nel file ~/compito/esercizio2/soluzione2.3_patch.txt.

4. ** Esegui il seguente esperimento: considera l'output del comando "**objdump -t init_list.o**". Ora metti "**static**" davanti a "**long int fib(long int f1)**" ed esegui ancora lo stesso comando. Mostra le differenze. Quale e' la semantica di questo uso di "**static**"?

Metti nel file ~/compito/esercizio2/soluzione2.4.txt i comandi per portare a termine l'esperimento (fai copia-e-incolla dal terminale) e la risposta alla domanda.

5. ** Considera una esecuzione di fib con parametro 15. Mostra il contenuto dello stack all'inizio della 2000-esima esecuzione della funzione fib().

Scrivi nel file ~/compito/esercizio2/soluzione2.5.txt la sessione di debug (fai copia-e-incolla dal terminale).