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

Sistemi Operativi 1 — A.A. 2004-2005, prova pratica del 26 aprile 2005

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.

Esercizio 0

- Scarica dall'url http://192.168.161.70/compito_a.tgz 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_a.tgz ; tar xvzf compito_a.tgz”)
- 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.

Esercizio 1

Il file di testo bgp_updates.txt contiene dati relativi a routing update BGP. Il file contiene record separati da linee vuote. I campi sono: NEXT_HOP, TIME, ASPATH, AGGREGATOR, MULTI_EXIT_DISC, WITHDRAW, ORIGIN, TYPE, ATOMIC_AGGREGATE, ANNOUNCE, TO, FROM, COMMUNITY. Su ciascuna linea c'è il nome del campo seguito da “:” e il suo valore. Ci sono tre eccezioni: ATOMIC_AGGREGATE non ha valore, ANNOUNCE e WITHDRAW hanno come valore una lista di prefissi IP separati su più linee, ma tali linee sono riconoscibili perché iniziano con due spazi. Tutti i campi sono presenti al più una volta in ciascun record. (Suggerimento: per il processamento di questo file tramite awk considera la possibilità di porre RS="", cioè stringa vuota, e considera inoltre la funzione gsub per effettuare sostituzioni)

1. *Quanti sono i record che contengono il campo ATOMIC_AGGREGATE?

Scrivi nel file esercizio1/soluzione1.1.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).

2. **Dai una lista degli indirizzi IP presenti nel campo FROM con a fianco il numero delle volte che ciascun prefisso appare in tale campo nel file.

Scrivi nel file esercizio1/soluzione1.2.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).

3. ***Dato un file F del tipo di bgp_updates.txt, considera l'insieme $X=\{h_1, h_2, ..., h_n\}$ (senza ripetizioni) dei valori che assume il campo NEXT_HOP in F. Scrivi uno script che dato sulla riga di comando il nome del file F produca un numero di files pari alla cardinalità di X, ciascuno di nome next_hop_h_i.txt, contenente i record di F per cui il campo NEXT_HOP è proprio h_i i record scritti in tali files devono essere privati del campo NEXT_HOP. (suggerimento: fai tante passate quanti sono i prefissi).

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

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

Sistemi Operativi 1 — A.A. 2004-2005, prova pratica del 26 aprile 2005

Esercizio 2

Il file esercizio2/check/tabella1.txt è organizzato in record i cui campi sono id (intero), superiore (intero), nome (stringa). Il valore di id deve identificare univocamente il record, il valore di superiore deve far riferimento ad un id presente nel file. Il progetto esercizio2/check dovrebbe produrre un programma che, dato un file come tabella1.txt in standard input, carichi i record in una lista in memoria e verifichi se tutti gli id sono univoci e se tutti i valori del campo superiore fanno riferimento ad id nel file. Putroppo di tale progetto si ha solo un file esercizio2/check/main.c che contiene degli errori. Svolgi le seguenti attività per rimettere a posto il progetto.

1. *Compila e mostra gli errori di compilazione
Scrivi nel file esercizio2/soluzione2.1.txt il comando usato per compilare e l'output prodotto

2. *Correggi gli errori di compilazione e mostra che la compilazione va a buon fine.
Scrivi nel file esercizio2/soluzione2.2.txt il comando usato per compilare e l'output prodotto

3. *Dividi il progetto in più file .c, producendo anche i relativi include files .h.
Metti i file all'interno della directory esercizio2, dai a tali file i nomi che preferisci

4. *crea la patch per il progetto check contenente le modifiche effettuate fino a questo punto del lavoro e mostra i comandi per generarla e applicarla
La patch si deve chiamare esercizio2/check.patch, scrivi le descrizioni dei comandi per la generazione e l'applicazione della patch in esercizio2/soluzione2.4.txt

5. **Crea un Makefile con i seguenti target (possibilmente avvalendoti delle regole predefinite di make
   - checksup_dyn: eseguibile linkato dinamicamente con simboli di debug
   - checksup_stat: eseguibile linkato staticamente senza simboli di debug
   - clean: cancella tutti i file .o
   - delete: cancella l'eseguibile e i file .o (dipende da clean)
   - test: esegue checksup_dyn su tabella1.txt 4 volte, rispettivamente con parametro 0, 1, 2 e 3
6. *Esegui il programma checksup_dyn su tabella1.txt mostrando che il risultato per il check dei superiori non è corretto.
Scrivi nel file esercizio2/soluzione2.6.txt i comandi usati per la compilazione e l'esecuzione e l'output prodotto e i tuoi commenti.

7. ***Individua il problema con l'aiuto del debugger
Riporta nel file esercizio2/soluzione2.7.txt la sessione di debugging.

8. *Correggi il programma e mostra l'output corretto
Riporta nel file esercizio2/soluzione2.8.txt l'output del programma corretto.

Istruzioni per la consegna del compito

Non spegnere il calcolatore. Recati dal docente con questo foglio compilato.

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

Sistemi Operativi 1 — A.A. 2004-2005, prova pratica del 26 aprile 2005

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.

Esercizio 0

- Scarica dall'url http://192.168.161.70/compito_b.tgz 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_b.tgz ; tar xvzf compito_b.tgz")
- 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.

Esercizio 1

Il file di testo bgp_updates.txt contiene dati relativi a routing update BGP. Il file contiene record separati da linee vuote. I campi sono: NEXT_HOP, TIME, ASPATH, AGGREGATOR, MULTI_EXIT_DISC, WITHDRAW, ORIGIN, TYPE, ATOMIC_AGGREGATE, ANNOUNCE, TO, FROM, COMMUNITY. Su ciascuna linea c'è il nome del campo seguito da ":" e il suo valore. Ci sono tre eccezioni: ATOMIC_AGGREGATE non ha valore, ANNOUNCE e WITHDRAW hanno come valore una lista di prefissi IP separati su più linee, ma tali linee sono riconoscibili perché iniziano con due spazi. Tutti i campi sono presenti al più una volta in ciascun record. (Suggerimento: per il processamento di questo file tramite awk considera la possibilità di porre RS="", cioè stringa vuota, e considera inoltre la funzione gsub per effettuare sostituzioni)

1. *Quanti sono i record che contengono il campo COMMUNITY?

Scrivi nel file esercizio1/soluzione1.1.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).

2. **Dai una lista dei prefissi presenti nei campi ANNOUNCE e WITHDRAW con a fianco il numero delle volte che ciascun prefisso appare in tali campi nel file.

Scrivi nel file esercizio1/soluzione1.2.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).

3. ***Una *community* è una coppia di numeri (es. 1668:31000). Le community sono presenti all'interno del campo COMMUNITY separate da spazi. Dato un file F del tipo di bgp_updates.txt, considera l'insieme $X=\{a_1, a_2, ..., a_n\}$ (senza ripetizioni) dei numeri per cui esiste una community $a_i:b_i$ citata in F. Scrivi uno script che dato sulla riga di comando il nome del file F produca un numero di files pari alla cardinalità di X, ciascuno di nome community_a_i.txt, contenente i record di F per cui il campo COMMUNITY contenga almeno una community $a_i:b_i$. (suggerimento: fai tante passate quanti sono le community presenti in F).

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

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

Sistemi Operativi 1 — A.A. 2004-2005, prova pratica del 26 aprile 2005

Esercizio 2

Il file esercizio2/check/tabella1.txt è organizzato in record i cui campi sono id (intero), superiore (intero), nome (stringa). Il valore di id deve identificare univocamente il record, il valore di superiore deve far riferimento ad un id presente nel file. Il progetto esercizio2/check dovrebbe produrre un programma che, dato un file come tabella1.txt in standard input, carichi i record in una lista in memoria e verifichi se tutti gli id sono univoci e se tutti i valori del campo superiore fanno riferimento ad id nel file. Putroppo di tale progetto si ha solo un file esercizio2/check/main.c che contiene degli errori. Svolgi le seguenti attività per rimettere a posto il progetto.

1. *Compila e mostra gli errori di compilazione
   Scrivi nel file esercizio2/soluzione2.1.txt il comando usato per compilare e l'output prodotto
2. *Correggi gli errori di compilazione e mostra che la compilazione va a buon fine.
   Scrivi nel file esercizio2/soluzione2.2.txt il comando usato per compilare e l'output prodotto
3. *Dividi il progetto in più file .c, producendo anche i relativi include files .h.
   Metti i file all'interno della directory esercizio2, dai a tali file i nomi che preferisci
4. *Crea la patch per il progetto check contenente le modifiche effettuate fino a questo punto del lavoro e mostra i comandi per generarla e applicarla
   La patch si deve chiamare esercizio2/check.patch, scrivi le descrizioni dei comandi per la generazione e l'applicazione della patch in esercizio2/soluzione2.4.txt
5. **Crea un Makefile con i seguenti target (possibilmente avvalendoti delle regole predefinite di make
   - dcheck: eseguibile linkato dinamicamente con simboli di debug
   - scheck: eseguibile linkato staticamente senza simboli di debug
   - clean: cancella tutti i file .o
   - delete: cancella l'eseguibile e i file .o (dipende da clean)
   - checksup.tar.gz: pacchetto contenente i soli file sorgenti e questo makefile (dipende da delete)
6. *Esegui il programma dcheck su tabella1.txt mostrando che i risultati dei check sono errati.
   Scrivi nel file esercizio2/soluzione2.6.txt i comandi usati per la compilazione e l'esecuzione e l'output prodotto e i tuoi commenti.
7. ***Individua il problema con l'aiuto del debugger
   Riporta nel file esercizio2/soluzione2.7.txt la sessione di debugging.
8. Correggi il programma e mostra l'output corretto
   Riporta nel file esercizio2/soluzione2.8.txt l'output del programma corretto.

Istruzioni per la consegna del compito

Non spegnere il calcolatore. Recati dal docente con questo foglio compilato.

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

Sistemi Operativi 1 — A.A. 2004-2005, prova pratica del 26 aprile 2005

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.

Esercizio 0

- Scarica dall'url http://192.168.161.70/compito_c.tgz 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_c.tgz ; tar xvzf compito_c.tgz")
- 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.

Esercizio 1

Il file di testo bgp_updates.txt contiene dati relativi a routing update BGP. Il file contiene record separati da linee vuote. I campi sono: NEXT_HOP, TIME, ASPATH, AGGREGATOR, MULTI_EXIT_DISC, WITHDRAW, ORIGIN, TYPE, ATOMIC_AGGREGATE, ANNOUNCE, TO, FROM, COMMUNITY. Su ciascuna linea c'è il nome del campo seguito da ":" e il suo valore. Ci sono tre eccezioni: ATOMIC_AGGREGATE non ha valore, ANNOUNCE e WITHDRAW hanno come valore una lista di prefissi IP separati su più linee, ma tali linee sono riconoscibili perché iniziano con due spazi. Tutti i campi sono presenti al più una volta in ciascun record. (Suggerimento: per il processamento di questo file tramite awk considera la possibilità di porre RS="", cioè stringa vuota, e considera inoltre la funzione gsub per effettuare sostituzioni)

1. *Quanti sono i record che contengono il campo ANNOUNCE?

Scrivi nel file esercizio1/soluzione1.1.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).

2. **Una *community* è una coppia di numeri (es. 1668:31000). Nel campo COMMUNITY le community sono separate da spazi. Dai la lista delle community citate nel file con a fianco il numero di volte che ciascuna community appare.

Scrivi nel file esercizio1/soluzione1.2.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).

3. ***Un *as-number* è un numero (es. 1668). Nel campo ASPATH gli as-number sono separati da spazi. In un ASPATH l'as-number che compare a destra di tutti gli altri è detto *originator* (es. per 11 47 129 88 25 l'originator è 25). Dato un file F del tipo di bgp_updates.txt, considera l'insieme $X=\{a_1, a_2, ..., a_n\}$ degli originator che appaiono in F. Scrivi uno script che dato sulla riga di comando il nome del file F produca un numero di files pari alla cardinalità di X, ciascuno di nome originator_a_i.txt, contenente i record di F per cui il campo ASPATH contenga a_i come originator. (suggerimento: fai tante passate quanti sono i possibili originator, considera in awk la variabile NF).

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

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

Sistemi Operativi 1 — A.A. 2004-2005, prova pratica del 26 aprile 2005

Esercizio 2

Il file esercizio2/check/tabella1.txt è organizzato in record i cui campi sono id (intero), superiore (intero), nome (stringa). Il valore di id deve identificare univocamente il record, il valore di superiore deve far riferimento ad un id presente nel file. Il progetto esercizio2/check dovrebbe produrre un programma che, dato un file come tabella1.txt in standard input, carichi i record in una lista in memoria e verifichi se tutti gli id sono univoci e se tutti i valori del campo superiore fanno riferimento ad id nel file. Putroppo di tale progetto si ha solo un file esercizio2/check/main.c che contiene degli errori. Svolgi le seguenti attività per rimettere a posto il progetto.

1. *Compila e mostra gli errori di compilazione
   Scrivi nel file esercizio2/soluzione2.1.txt il comando usato per compilare e l'output prodotto
2. *Correggi gli errori di compilazione e mostra che la compilazione va a buon fine.
   Scrivi nel file esercizio2/soluzione2.2.txt il comando usato per compilare e l'output prodotto
3. *Dividi il progetto in più file .c, producendo anche i relativi include files .h.
   Metti i file all'interno della directory esercizio2, dai a tali file i nomi che preferisci
4. *Crea la patch per il progetto check contenente le modifiche effettuate fino a questo punto del lavoro e mostra i comandi per generarla e applicarla
   La patch si deve chiamare esercizio2/check.patch, scrivi le descrizioni dei comandi per la generazione e l'applicazione della patch in esercizio2/soluzione2.4.txt
5. **Crea un Makefile con i seguenti target (possibilmente avvalendoti delle regole predefinite di make
   - checksup_dyn: eseguibile linkato dinamicamente con simboli di debug
   - checksup_stat: eseguibile linkato staticamente senza simboli di debug
   - clean: cancella tutti i file .o
   - delete: cancella l'eseguibile e i file .o (dipende da clean)
   - test: esegue checksup_dyn su tabella1.txt 4 volte, rispettivamente con parametro 0, 1, 2 e 3
6. *Esegui il programma checksup_dyn su tabella1.txt mostrando che il risultato per il check dei superiori non è corretto.
   Scrivi nel file esercizio2/soluzione2.6.txt i comandi usati per la compilazione e l'esecuzione e l'output prodotto e i tuoi commenti.
7. ***Individua il problema con l'aiuto del debugger
   Riporta nel file esercizio2/soluzione2.7.txt la sessione di debugging.
8. *Correggi il programma e mostra l'output corretto
   Riporta nel file esercizio2/soluzione2.8.txt l'output del programma corretto.

Istruzioni per la consegna del compito

Non spegnere il calcolatore. Recati dal docente con questo foglio compilato.

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

Sistemi Operativi 1 — A.A. 2004-2005, prova pratica del 26 aprile 2005

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.

Esercizio 0

- Scarica dall'url http://192.168.161.70/compito_d.tgz 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_d.tgz ; tar xvzf compito_d.tgz”)
- 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.

Esercizio 1

Il file di testo bgp_updates.txt contiene dati relativi a routing update BGP. Il file contiene record separati da linee vuote. I campi sono: NEXT_HOP, TIME, ASPATH, AGGREGATOR, MULTI_EXIT_DISC, WITHDRAW, ORIGIN, TYPE, ATOMIC_AGGREGATE, ANNOUNCE, TO, FROM, COMMUNITY. Su ciascuna linea c'è il nome del campo seguito da “:” e il suo valore. Ci sono tre eccezioni: ATOMIC_AGGREGATE non ha valore, ANNOUNCE e WITHDRAW hanno come valore una lista di prefissi IP separati su più linee, ma tali linee sono riconoscibili perché iniziano con due spazi. Tutti i campi sono presenti al più una volta in ciascun record. (Suggerimento: per il processamento di questo file tramite awk considera la possibilità di porre RS="", cioè stringa vuota, e considera inoltre la funzione gsub per effettuare sostituzioni)

1. *Quanti sono i record che contengono il campo WITHDRAW?

Scrivi nel file esercizio1/soluzione1.1.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).

2. **Un as-number è un numero (es. 1668). Nel campo ASPATH gli as-number sono separati da spazi. Dai la lista di tutti gli as-number presenti nel campo ASPATH con a fianco il numero di volte che ciascun as-number appare.

Scrivi nel file esercizio1/soluzione1.2.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).

3. ***In un ASPATH il primo as-number è detto *peer* (es. per 11 47 129 88 25 il peer è 11). Dato un file F del tipo di bgp_updates.txt, considera l'insieme $X=\{a_1, a_2, ..., a_n\}$ (senza ripetizioni) dei peer che appaiono in F. Scrivi uno script che dato sulla riga di comando il nome del file F produca un numero di files pari alla cardinalità di X, ciascuno di nome peer_a_i.txt, contenente i record di F per cui il campo ASPATH contenga a_i come peer. Inoltre si vuole che i record nei file peer_a_i.txt siano privati del campo NEXT_HOP. (suggerimento: fai tante passate quanti sono peer).

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

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

Sistemi Operativi 1 — A.A. 2004-2005, prova pratica del 26 aprile 2005

Esercizio 2

Il file esercizio2/check/tabella1.txt è organizzato in record i cui campi sono id (intero), superiore (intero), nome (stringa). Il valore di id deve identificare univocamente il record, il valore di superiore deve far riferimento ad un id presente nel file. Il progetto esercizio2/check dovrebbe produrre un programma che, dato un file come tabella1.txt in standard input, carichi i record in una lista in memoria e verifichi se tutti gli id sono univoci e se tutti i valori del campo superiore fanno riferimento ad id nel file. Putroppo di tale progetto si ha solo un file esercizio2/check/main.c che contiene degli errori. Svolgi le seguenti attività per rimettere a posto il progetto.

1. *Compila e mostra gli errori di compilazione
   Scrivi nel file esercizio2/soluzione2.1.txt il comando usato per compilare e l'output prodotto
2. *Correggi gli errori di compilazione e mostra che la compilazione va a buon fine.
   Scrivi nel file esercizio2/soluzione2.2.txt il comando usato per compilare e l'output prodotto
3. *Dividi il progetto in più file .c, producendo anche i relativi include files .h.
   Metti i file all'interno della directory esercizio2, dai a tali file i nomi che preferisci
4. *Crea la patch per il progetto check contenente le modifiche effettuate fino a questo punto del lavoro e mostra i comandi per generarla e applicarla
   La patch si deve chiamare esercizio2/check.patch, scrivi le descrizioni dei comandi per la generazione e l'applicazione della patch in esercizio2/soluzione2.4.txt
5. **Crea un Makefile con i seguenti target (possibilmente avvalendoti delle regole predefinite di make
   - dcheck: eseguibile linkato dinamicamente con simboli di debug
   - scheck: eseguibile linkato staticamente senza simboli di debug
   - clean: cancella tutti i file .o
   - delete: cancella l'eseguibile e i file .o (dipende da clean)
   - checksup.tar.gz: pacchetto contenente i soli file sorgenti e questo makefile (dipende da delete)
6. *Esegui il programma dcheck su tabella1.txt mostrando che i risultati dei check sono errati.
   Scrivi nel file esercizio2/soluzione2.6.txt i comandi usati per la compilazione e l'esecuzione e l'output prodotto e i tuoi commenti.
7. ***Individua il problema con l'aiuto del debugger
   Riporta nel file esercizio2/soluzione2.7.txt la sessione di debugging.
8. *Correggi il programma e mostra l'output corretto
   Riporta nel file esercizio2/soluzione2.8.txt l'output del programma corretto.

Istruzioni per la consegna del compito

Non spegnere il calcolatore. Recati dal docente con questo foglio compilato.

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

Sistemi Operativi 1 — A.A. 2004-2005, prova pratica del 26 aprile 2005

Compito E

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.

Esercizio 0

- Scarica dall'url http://192.168.161.70/compito_e.tgz 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_e.tgz ; tar xvzf compito_e.tgz”)
- 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.

Esercizio 1

Il file di testo bgp_updates.txt contiene dati relativi a routing update BGP. Il file contiene record separati da linee vuote. I campi sono: NEXT_HOP, TIME, ASPATH, AGGREGATOR, MULTI_EXIT_DISC, WITHDRAW, ORIGIN, TYPE, ATOMIC_AGGREGATE, ANNOUNCE, TO, FROM, COMMUNITY. Su ciascuna linea c'è il nome del campo seguito da “:” e il suo valore. Ci sono tre eccezioni: ATOMIC_AGGREGATE non ha valore, ANNOUNCE e WITHDRAW hanno come valore una lista di prefissi IP separati su più linee, ma tali linee sono riconoscibili perché iniziano con due spazi. Tutti i campi sono presenti al più una volta in ciascun record. (Suggerimento: per il processamento di questo file tramite awk considera la possibilità di porre RS="", cioè stringa vuota, e considera inoltre la funzione gsub per effettuare sostituzioni)

1. *Quanti sono i record che hanno il campo FROM con valore “134.55.200.31 AS293” ?

Scrivi nel file esercizio1/soluzione1.1.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).

2. **Considera i prefissi presenti nei campi ANNOUNCE e WITHDRAW. Ciascuno di essi ha una lunghezza espressa con un numero dopo la barra (es. 193.204.161.0/24, 24 è la lunghezza). Dai una lista delle lunghezze dei prefissi presenti nei campi ANNOUNCE e WITHDRAW con a fianco il numero delle volte che ciascuna lunghezza appare in tali campi nel file.

Scrivi nel file esercizio1/soluzione1.2.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).

3. ***Considera un file F del tipo di bgp_updates.txt. Un prefisso p compare in F un certo numero di volte come parte di un campo ANNOUNCE o WITHDRAW, chiama $n(p)$ tale numero. Scrivi uno script che dato sulla riga di comando il nome del file F produca in output un file F.stat che per ogni prefisso p che appare in un campo ANNOUNCE o WITHDRAW scriva affianco ad esso $n(p)$, esempio 193.204.161.0/24 può diventare 193.204.161.0/24 15 (suggerimento fai tante passate quanti sono i prefissi che appaiono nei campi ANNOUNCE).

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

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

Sistemi Operativi 1 — A.A. 2004-2005, prova pratica del 26 aprile 2005

Esercizio 2

Il file esercizio2/check/tabella1.txt è organizzato in record i cui campi sono id (intero), superiore (intero), nome (stringa). Il valore di id deve identificare univocamente il record, il valore di superiore deve far riferimento ad un id presente nel file. Il progetto esercizio2/check dovrebbe produrre un programma che, dato un file come tabella1.txt in standard input, carichi i record in una lista in memoria e verifichi se tutti gli id sono univoci e se tutti i valori del campo superiore fanno riferimento ad id nel file. Putroppo di tale progetto si ha solo un file esercizio2/check/main.c che contiene degli errori. Svolgi le seguenti attività per rimettere a posto il progetto.

1. *Compila e mostra gli errori di compilazione
   Scrivi nel file esercizio2/soluzione2.1.txt il comando usato per compilare e l'output prodotto
2. *Correggi gli errori di compilazione e mostra che la compilazione va a buon fine.
   Scrivi nel file esercizio2/soluzione2.2.txt il comando usato per compilare e l'output prodotto
3. *Dividi il progetto in più file .c, producendo anche i relativi include files .h.
   Metti i file all'interno della directory esercizio2, dai a tali file i nomi che preferisci
4. *Crea la patch per il progetto check contenente le modifiche effettuate fino a questo punto del lavoro e mostra i comandi per generarla e applicarla
   La patch si deve chiamare esercizio2/check.patch, scrivi le descrizioni dei comandi per la generazione e l'applicazione della patch in esercizio2/soluzione2.4.txt
5. **Crea un Makefile con i seguenti target (possibilmente avvalendoti delle regole predefinite di make
   - chk1: eseguibile linkato dinamicamente con simboli di debug, alla fine della generazione dell'eseguibile esegue un comando che mostra le librerie che l'eseguibile linca dinamicamente.
   - chk2: eseguibile linkato staticamente senza simboli di debug. Alla fine della generazione dell'eseguibile esegue un comando che mostra che tale eseguibile è effettivamente linkato staticamente
   - clean: cancella tutti i file .o
   - delete: cancella l'eseguibile e i file .o (dipende da clean)
   - test: esegue chk1 solo sulle prime 10 righe di tabella1.txt per 4 volte, rispettivamente con parametro 0, 1, 2 e 3
6. *Esegui il programma chk1 su tabella1.txt mostrando che entrambi i check danno risultato errato.
   Scrivi nel file esercizio2/soluzione2.6.txt i comandi usati per la compilazione e l'esecuzione e l'output prodotto e i tuoi commenti.
7. ***Individua il problema con l'aiuto del debugger
   Riporta nel file esercizio2/soluzione2.7.txt la sessione di debugging.
8. *Correggi il programma e mostra l'output corretto
   Riporta nel file esercizio2/soluzione2.8.txt l'output del programma corretto.

Istruzioni per la consegna del compito

Non spegnere il calcolatore. Recati dal docente con questo foglio compilato.

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

Sistemi Operativi 1 — A.A. 2004-2005, prova pratica del 26 aprile 2005

Compito F

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.

Esercizio 0

- Scarica dall'url http://192.168.161.70/compito_f.tgz 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_f.tgz ; tar xvzf compito_f.tgz")
- 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.

Esercizio 1

Il file di testo bgp_updates.txt contiene dati relativi a routing update BGP. Il file contiene record separati da linee vuote. I campi sono: NEXT_HOP, TIME, ASPATH, AGGREGATOR, MULTI_EXIT_DISC, WITHDRAW, ORIGIN, TYPE, ATOMIC_AGGREGATE, ANNOUNCE, TO, FROM, COMMUNITY. Su ciascuna linea c'è il nome del campo seguito da ":" e il suo valore. Ci sono tre eccezioni: ATOMIC_AGGREGATE non ha valore, ANNOUNCE e WITHDRAW hanno come valore una lista di prefissi IP separati su più linee, ma tali linee sono riconoscibili perché iniziano con due spazi. Tutti i campi sono presenti al più una volta in ciascun record. (Suggerimento: per il processamento di questo file tramite awk considera la possibilità di porre RS="", cioè stringa vuota, e considera inoltre la funzione gsub per effettuare sostituzioni)

1. *Quanti sono i record che hanno il campo TO con valore "198.32.162.102 AS6447"?

Scrivi nel file esercizio1/soluzione1.1.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).

2. **Un as-number è un numero (es. 1668). Nel campo ASPATH gli as-number sono separati da spazi. In un ASPATH l'as-number che compare a destra di tutti gli altri è detto *originator* (es. per 11 47 129 88 25 l'originator è 25). Dai la lista di tutti gli originator con a fianco il numero di volte che ciascun originator appare come tale nel file (suggerimento considera la variabile NF di awk).

Scrivi nel file esercizio1/soluzione1.2.txt il comando usato e il suo output (fai copia-e-incolla dal terminale).

3. ***Considera un file F del tipo di bgp_updates.txt. Un *as-number* è un numero (es. 1668). Nel campo ASPATH di F gli as-number sono separati da spazi. Ciascun as-number p comapre $n(p)$ volte nei campi ASPATH di F. Scrivi uno script che dato sulla riga di comando il nome del file F produca in output un file F.stat che per ogni as-number p che appare in un campo ASPATH di F scriva affianco ad esso, tra parentesi, $n(p)$. Ad esempio ASPATH: 4 5 3 2 potrebbe diventare ASPATH: 4 (10) 5 (28) 3 (4) 2 (25). (suggerimento fai tante passate quanti sono gli as-number presenti in F).

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

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

Sistemi Operativi 1 — A.A. 2004-2005, prova pratica del 26 aprile 2005

Esercizio 2

Il file esercizio2/check/tabella1.txt è organizzato in record i cui campi sono id (intero), superiore (intero), nome (stringa). Il valore di id deve identificare univocamente il record, il valore di superiore deve far riferimento ad un id presente nel file. Il progetto esercizio2/check dovrebbe produrre un programma che, dato un file come tabella1.txt in standard input, carichi i record in una lista in memoria e verifichi se tutti gli id sono univoci e se tutti i valori del campo superiore fanno riferimento ad id nel file. Putroppo di tale progetto si ha solo un file esercizio2/check/main.c che contiene degli errori. Svolgi le seguenti attività per rimettere a posto il progetto.

1. *Compila e mostra gli errori di compilazione
Scrivi nel file esercizio2/soluzione2.1.txt il comando usato per compilare e l'output prodotto

2. *Correggi gli errori di compilazione e mostra che la compilazione va a buon fine.
Scrivi nel file esercizio2/soluzione2.2.txt il comando usato per compilare e l'output prodotto

3. *Dividi il progetto in più file .c, producendo anche i relativi include files .h.
Metti i file all'interno della directory esercizio2, dai a tali file i nomi che preferisci

4. *Crea la patch per il progetto check contenente le modifiche effettuate fino a questo punto del lavoro e mostra i comandi per generarla e applicarla
La patch si deve chiamare esercizio2/check.patch, scrivi le descrizioni dei comandi per la generazione e l'applicazione della patch in esercizio2/soluzione2.4.txt

5. **Crea un Makefile con i seguenti target (possibilmente avvalendoti delle regole predefinite di make
   - controlla1: eseguibile linkato staticamente senza simboli di debug. Alla fine della generazione dell'eseguibile stampa la lunghezza del file generato
   - controlla2: eseguibile linkato dinamicamente con simboli di debug, alla fine della generazione dell'eseguibile stamapa la lunghezza del file generato
   - clean: cancella tutti i file .o
   - delete: cancella l'eseguibile e i file .o (dipende da clean)
   - test: esegue contrlla2 solo sulle ultime 20 righe di tabella1.txt per 4 volte, rispettivamente con parametro 0, 1, 2 e 3

6. *Esegui il programma controlla2 su tabella1.txt mostrando che il check sull'unicità degli id dà risultato errato.
Scrivi nel file esercizio2/soluzione2.6.txt i comandi usati per la compilazione e l'esecuzione e l'output prodotto e i tuoi commenti.

7. ***Individua il problema con l'aiuto del debugger
Riporta nel file esercizio2/soluzione2.7.txt la sessione di debugging.

8. *Correggi il programma e mostra l'output corretto
Riporta nel file esercizio2/soluzione2.8.txt l'output del programma corretto.

Istruzioni per la consegna del compito

Non spegnere il calcolatore. Recati dal docente con questo foglio compilato.