Laboratorio di Algoritmi: indicazioni utili per il progetto

Regole

1. Il progetto deve essere svolto **individualmente**.
2. Il progetto deve essere svolto **interamente**.

Organizzazione Il testo descrive un problema da risolvere in modo algoritmico. Fornisce indizi chiari, ma in genere non espliciti, sul tipo di oggetti matematici da usare per modellarlo, sugli algoritmi per risolverlo e sulle strutture dati per conservare dati e risultati. La capacità di scegliere gli strumenti corretti fa quindi parte della valutazione. Nei casi, più rari, in cui algoritmi e strutture dati sono esplicitamente indicati la valutazione si concentra sulla descrizione e sull'analisi degli algoritmi. Se si è tentati di seguire una strada diversa, si consiglia di contattarmi: potreste aver ragione, ma i suggerimenti di solito sono ben motivati.

Con il testo vengono forniti dati di esempio e soluzioni. Possono contenere degli errori (che vengono corretti appena segnalati) e in alcuni progetti è possibile che esistano soluzioni corrette molteplici (in genere, la cosa è segnalata nel testo). È però molto probabile che un codice corretto ottenga le soluzioni fornite. D'altra parte, ottenere le stesse soluzioni non garantisce l'assoluta correttezza del codice.

Valutazione Il progetto viene valutato in base a sei criteri, raccolti in due gruppi:

1. per quanto riguarda la relazione:
   (a) la completezza, cioè la descrizione di ogni aspetto rilevante, rivolta a un lettore che non conosce il tema del progetto, ma ha seguito corsi di algoritmi e programmazione;
   (b) la correttezza, cioè l'assenza di errori teorici e la corrispondenza fra quanto descritto nella relazione e quanto realizzato nel codice;
   (c) la struttura, cioè l'organizzazione degli argomenti in sezioni, il filo logico, la chiarezza espositiva, l'aspetto estetico;
2. per quanto riguarda il codice:
   (a) la correttezza, cioè l'assenza di errori (di algoritmi e di programmazione) nella lettura dei dati, nella gestione delle strutture dati, nell'esecuzione degli algoritmi, nella stampa dei risultati;
   (b) l'efficienza, cioè l'uso di strutture dati e algoritmi appropriati per risolvere il problema nel tempo minimo e/o con l'occupazione di memoria minima (privilegiando il tempo rispetto allo spazio, quando si debba scegliere);
   (c) la struttura, cioè l'organizzazione del codice in moduli, dei moduli in funzioni, delle funzioni in blocchi, l'uso delle variabili e l'adeguatezza dei commenti.

Relazione La relazione soddisfa il criterio di completezza quando è un documento compiuto, che descrive sinteticamente al lettore:

1. il problema a livello concettuale (senza entrare in dettagli tecnici come i formati di ingresso e uscita);

2. il modello astratto adottato e i motivi di tale scelta;

3. gli algoritmi e le strutture dati adottate, senza entrare nei dettagli forniti dal corso, ma illustrando i motivi di tali scelte e le valutazioni di complessità temporale e spaziale.

La relazione soddisfa il criterio di correttezza quando, scorrendola in parallelo al codice, si verifica pagina dopo pagina che le operazioni e le analisi di complessità descritte corrispondono a quelle implementate e ai risultati della teoria.

La relazione soddisfa il criterio di struttura se ha un'organizzazione "top-down" parallela a quella del codice, che permette di leggerla una volta sola dal principio alla fine, senza dover saltare avanti e indietro fra sezioni diverse. Questo non significa seguire istruzione per istruzione il codice, ma presentare prima il problema generale poi i sottoproblemi particolari, prima i concetti astratti poi la loro realizzazione concreta, rimandando i discorsi più dettagliati a sezioni successive, esattamente come il codice non contiene pagine di istruzioni tutte allo stesso livello, ma poche chiamate a procedure di livello inferiore che svolgono chiare funzioni ausiliarie, alle quali si passa dopo aver capito che cosa fa il codice nel suo complesso.

Le analisi di complessità temporale e spaziale di ogni funzione verranno naturalmente discusse subito dopo la descrizione delle operazioni, come ovvia conseguenza della descrizione stessa, e ricomposte al termine nella complessità dell'intero algoritmo. Non saranno riferite a un generico indice n, ma a ciascuna delle quantità che descrivono la dimensione dell'istanza. Conviene fare attenzione a banalizzare l'analisi ignorando degli indici o facendo ipotesi troppo semplificative, per non danneggiare la completezza e la correttezza della relazione.

Codice Conviene compilare il codice con le opzioni `-std=c89` e `-Wall` e rimediare agli avvertimenti così ottenuti, perché possono aiutare a scoprire errori sfuggenti. Può essere utile anche l'opzione `-pedantic`, anche se genera avvertimenti un po' eccessivi. Gli avvertimenti più tipici che emergono in questo modo riguardano la divisione tra parte dichiarativa ed esecutiva delle procedure (da mantenere, rispettivamente, al principio e al termine), le dichiarazioni di contatori nei cicli e le dichiarazioni di vettori statici con lunghezza espressa da una variabile (che dovrebbero essere invece vettori dinamici).

Si carichino i dati nel modo più semplice possibile, senza salti mortali che rischiano di rendere la lettura dipendente dallo specifico file. Durante la valutazione verranno usati anche dati diversi da quelli forniti insieme al testo del progetto

Si carichino i file da linea di comando; se il testo del progetto indica che un'istanza consiste in più file, si carichino i file nell'ordine indicato.

Si rispetti rigorosamente il formato di uscita indicato nel testo del progetto (maiuscole, minuscole, a capi, ecc...). Facendo stampare i risultati a video e redirigendoli su un file di testo (con `>` seguito dal nome del file) aiuta a confrontarli con le soluzioni disponibili. Inoltre, risolve alcuni problemi legati alla stampa di lettere accentate.

Perché tanta pignoleria? Perché valutare in pochi giorni relazioni e codici (a volte anche una trentina) richiede di compilare e lanciare in batteria tutti i codici su tutte le istanze. Se ogni codice reagisce a modo suo, il procedimento si incaglia e diventa difficile mantenere un metro uniforme di valutazione.

Perché limitarsi al C89? Ci sono motivi tecnici che sarebbe lungo discutere qui, ma il motivo principale è che si semplifica l'analisi di complessità per gli studenti e si semplifica la ricerca di errori per me.

A puro titolo informativo (nessuna pubblicità), valuto i progetti con l'ultima versione di Microsoft Visual Studio Express: è un compilatore estremamente pignolo, che segnala errori ignorati dal `gcc` e comunque li fa emergere durante l'esecuzione (è tipico che io riceva programmi che funzionano sulle macchine degli studenti, ma non sulla mia: c'è sempre dietro un errore.

Spedizione Relazione e codice vanno raccolti in un file compattato (in formato .ZIP o .RAR o .GZ) e spediti al mio indirizzo di posta elettronica di ateneo entro la mezzanotte del giorno indicato sul testo del progetto.

Prima di spedire, si verifichi che l'allegato non contenga file eseguibili o ambigui, perché il server di posta cancella questi allegati.

Di solito, quando ricevo relazione e codice spedisco un messaggio di conferma della ricezione. A seconda dei casi, il messaggio può essere quasi immediato o più ritardato, in genere comunque entro un paio di giorni.

In caso di ritardi sia nella conferma della ricezione sia nella pubblicazione del progetto (che esce circa un mese prima della data ufficiale dell'orale), conviene contattarmi via mail: potrebbero esserci stati problemi tecnici.