PROGRAMMAZIONE AVANZATA

Specifiche del progetto d'esame

- Progetto d'esame
- Codice
- Commenti e documentazione
- Consegna
- Valutazione del progetto
- Punteggio base, discussione e voto
- Proposte di progetti d'esame
- Suggerimenti conclusivi

Progetto d’esame

In cosa consiste

- Programma che risolve di un semplice problema (gioco, quiz, problema matematico, ecc.).
- Il problema è a scelta
- Ci sono dei **vincoli** da rispettare su codice, documentazione, consegna, descritti nelle prossime slide.

Come si svolge

- Individualmente o in gruppo (max 2 persone).
- Se fatto in gruppo, i contributi dei singoli devono essere **equi** e **facilmente individuabili**

Codice

1. È **obbligatorio** usare almeno un **contenitore** in modo **non banale** (es. un contenitore la cui dimensione dipende dall'input del programma). La classe `std::string` **non conta** come contenitore.
2. È **obbligatorio** definire almeno una **classe non banale** diversa da quelle viste durante il corso. La classe deve essere dotata di stato (mutabile o no) e fornire metodi corrispondenti a "operazioni sensate" su oggetti di quella classe.
3. È **obbligatorio** usare **nomi rappresentativi in italiano senza abbreviazioni** per tutte le classi e i metodi definiti, eventualmente usando il carattere di sottolineatura per separare le parole. **SÌ** `calcola_trasposta`, **SÌ** `stampa_soluzione`, **NO** `calc_trasp`, **NO** `printsol`, **NO** `Gatto` o `Animale` se la classe modella un pesce
4. Sono **consentiti** l'uso e la modifica delle classi fornite dal docente durante il corso, incluse le soluzioni degli esercizi. Queste classi non contribuiscono ad assolvere agli obblighi di cui ai punti 1 e 2 anche se modificate e sono esentate dagli obblighi di cui al punto 3.
5. È **vietata** l'inclusione di qualsiasi codice non prodotto dagli autori del progetto o dal docente.

Commenti e documentazione

- È **obbligatorio** includere un file `LEGGIMI.txt` in formato testuale (`.txt`) o PDF (`.pdf`) o Markdown (`.md`) con le seguenti informazioni:
  1. **Descrizione del problema** risolto dal programma (1 paragrafo)
  2. **Descrizione sommaria** delle classi (1 frase per classe)
  3. **Indicazioni sull'uso del programma**, es. argomenti da passare dalla riga di comando, formato di file di input, ecc.
  4. Un **esempio d'uso** del programma con input fornito e output ottenuto
  5. Eventuali **riferimenti** a siti (es. Wikipedia) che descrivono sommariamente il problema, l'algoritmo, la tecnica utilizzata, ecc.
- È **vietato** l'uso di altri formati (es. RTF, DOC) per il file `LEGGIMI`
- È **obbligatorio** iniziare ogni file `.cc` con un commento che elenca tutti gli **autori** del progetto (nome, cognome, email, numero di matricola).
- È **obbligatorio** commentare ogni **classe** descrivendone brevemente lo scopo (= cosa rappresenta) e i campi (= come la rappresenta).
- Se presenti, gli **invarianti di classe** vanno indicati in commenti nei pressi dei campi della classe.
- È **obbligatorio** commentare ogni **metodo** descrivendone brevemente la funzione, gli argomenti ed il valore restituito, laddove presenti.

Consegna

- Il codice sorgente del progetto deve essere consegnato **una settimana prima** dell'appello d'esame. Il docente predisporrà un'apposita attività per la consegna nella pagina Moodle del corso.
- Il codice sorgente deve essere consegnato in un archivio dal nome `COGNOME_MATRICOLA.zip`. Il nome dell'archivio deve essere formato dal proprio **cognome** (privato di spazi ed eventuali apostrofi) seguito dal carattere `_` e infine dal proprio **numero di matricola**.
- Nel caso di progetto svolto in gruppo (max 2 persone), il caricamento deve essere effettuato da **un solo membro** eletto a rappresentante del gruppo seguendo le indicazioni del punto precedente per quanto riguarda il nome dell'archivio.
- L'archivio consegnato **deve contenere** tutti i file necessari per la compilazione e l'esecuzione del programma anche "offline" (senza una connessione Internet).
- L'archivio consegnato **non deve contenere** file generati dal proprio compilatore (`.exe`, `.obj`, `.o`, `.out`, ecc.).
- Se il programma necessita di accorgimenti particolari per la compilazione, indicarli in una nota nel file `LEGGIMI`. Il docente contatterà gli autori del progetto nel caso in cui riscontrasse difficoltà nella compilazione.

Valutazione del progetto

Il progetto viene valutato tenendo conto dei seguenti criteri, indicativamente elencati in ordine descrescente di importanza:

1. Uso appropriato delle **strutture dati** (classi, contenitori) e dei **comandi di controllo** (cicli, `if`, `return`, `throw`)
2. **Ordine** e **indentazione** del codice
3. Rispetto dei **vincoli** indicati in queste slide
4. **Correttezza**: il programma risolve correttamente il problema per tutti gli input validi (cf. punti 1 e 4 del file `LEGGIMI`)
5. Originalità e difficoltà del problema

Morali

- Meglio risolvere bene un problema semplice piuttosto che risolvere male un problema difficile.
- Un progetto ben fatto internamente con qualche difetto di esecuzione vale più di uno che funziona perfettamente ma è strutturato male

Punteggio base, discussione e voto

- Il docente assegna un **punteggio base** espresso in trentesimi a ogni progetto consegnato e ne riassume la valutazione in un **giudizio**.
- La **discussione** avviene:
  - se il docente lo ritiene necessario, oppure
  - su richiesta di chi desidera migliorare il punteggio base
- La discussione è sempre **individuale** anche se il progetto è fatto in gruppo. La discussione mira a verificare:
  - una buona padronanza di tutti i **concetti** e **strumenti di programmazione** presentati durante il corso
  - la perfetta conoscenza di **tutto** il codice del progetto (non solo della propria parte)
  - una buona capacità di apportare al volo semplici **modifiche** al codice del progetto (non solo sulla propria parte)
- La discussione può avere **qualsiasi esito**:
  - mancato superamento dell'esame (la discussione va ripetuta)
  - superamento dell'esame con **voto** pari, superiore o inferiore al punteggio base
- Se la discussione non avviene, il punteggio base diventa il voto finale.

Proposte di progetti d'esame

- Mastermind, Battaglia navale, Impiccato, Forza 4, Nim, Campo minato
- Inversione di una matrice
- Risoluzione di un sistema di equazioni lineari
- Analisi numerica (es. integrazione con metodo dei trapezi)
- Ricerca degli anagrammi di una parola usando il dizionario di italiano
- Ricerca di parole "vicine" a una parola data usando la distanza di Levenshtein
- Generazione automatica di cruciverba

Note

- Nel caso di giochi, non è necessario che il programma "sappia giocare". È sufficiente che il programma fornisca gli elementi essenziali del gioco o sappia controllare chi ha vinto, lasciando all'utilizzatore il controllo sulla strategia di gioco.
- Queste sono proposte di progetti **individuali**. In un gruppo di n persone, mi aspetto un progetto indicativamente n volte più corposo/complesso

Suggerimenti conclusivi

- Scegliete un **progetto che vi diverte/interessa** (più di altri) e magari che riguarda un problema con cui avete già familiarità. In questo modo potrete farvi subito un'idea abbastanza precisa di quanto possa essere facile/difficile affrontarlo tenendo conto dei vincoli descritti in precedenza.
- Non ha senso studiare a priori tutta la libreria standard del C++, che è vastissima. Se avete necessità di un'operazione (su stringhe, su contenitori, per fare input/output, ...) che non è troppo specifica del vostro programma, è probabile che questa operazione sia già disponibile sotto forma di metodo o algoritmo nella libreria standard. **Consultate la libreria all'occorrenza**, è questo il modo normale di utilizzarla.
- Esclusi i tre giorni precedenti ogni appello d'esame, sono disponibile per fare **ricevimento online** non solo per dubbi sui contenuti del corso, ma anche per discutere del progetto e consigliarvi se rimanete bloccati su qualche problema.

BUON LAVORO!