Domande orale Linguaggi di programmazione 07-06-2022

Domande modulo 1 (Gorrieri)

(Le stesse della scheda consegnata agli studenti, ma in modo puntiglioso).

Domande modulo 2 (Gabbrielli)

- Esistono linguaggi non tipati, se sì le sembra un'idea ragionevole, se no perchè?
  - che vantaggi ci sono per essere non tipato?
  - dal punto di vista dell'efficienza cambia qualcosa?
- Il tipo `void` cos'è?
  - che insieme di valori denota? > un insieme singoletto (secondo Gabbrielli, non secondo Giallorenzo)

Domande modulo 3 (Giallorenzo)

- Differenza tra manifest e infert typing
  - Pro e contro di infert e manifest typing
- La relazione che c'è tra sottotipaggio ed ereditarietà nell'orientamento ad oggetti
  - cosa ci permette di sapere il fatto che uno è sottotipo di un altro?
  - si vede che c'è una differenza pratica tra sottotipi e 'sottoclassi'? Se non definisce una classe, cosa definisce? > un'interfaccia
- Che differenza c'è tra array unidimensionali e multidimensionali?
  - il fatto di avere una row manage o una column manage cosa ci implica?
  - qual'è il pro di avere uno rispetto all'altro e perchè?
  - è vero dire che questi array non sono staticamente definiti? In che modo sono memorizzati?
  - array dinamici
- Il reference types
  - Il tipo più generico in java > objects
  - tutto quello che viene dopo è più specifico. Dunque quale può essere un'implementazione di reference types?
  - che differenza c'è tra maybe e option?
- Quali sono i vantaggi e svantaggi del fatto che alcuni linguaggi permettono di avere interoperabilità tra gli array e i puntatori? [2]

  - svantaggi e vantaggi dell'interoperabilità tra gli array e i puntatori, e fare un esempio
  - cosa significa che un puntatore diventa dangling?
  - se non avessimo l'interoperabilità e avessimo solo gli array, potremmo fare un controllo statico? Oppure si possono fare solo controlli dinamici?
  - si può dire in maniera statica il risultato di un'operazione?

- Cos'è il garbage e il garbage collection? Quali sono i metodi di implementazione? [4]
  - cosa possiamo utilizzare per la detection e la collection?
  - è possibile avere dangling pointers con la garbage collection?
  - cosa vuol dire avere la garbage collection? Che ruolo ha con i dangling reference/dangling pointers?
  - una vera alternativa? Quali sono i pro e i contro con gli altri? > reference count
    * che limitazioni ha?

- Quali sono le tecniche di protezione della memoria?
  - qual è uno dei grandi problemi di queste tecniche?

- Differenza tra tipaggio nominale e strutturale. [2]
  - perchè scelgo uno o l'altro?
  - sull'efficienza cosa si intende?
  - riesce a fare un esempio?
  - quali sono i pro e i contro?

- Tipaggio statico e tipaggio dinamico, differenze ed esempi di linguaggi.
  - che tipo di linguaggio è JavaScript?

- Cos'è un tipo prodotto e fare un esempio come struttura.
  - differenza tra record e array
  - cosa serve per essere un `record`?
  - se non avessimo la possibilità di dare dei campi cosa avremmo al posto dei `record`? > tuple

- Polimorfismo, quali sono e le differenze.
  - come si chiama la differenza tra il nome del tipo di polimorfismo e il polimorfismo vero e proprio? > parametro specificato quando istanziamo un tipo specificato
  - questa direzione viene sempre mantenuta o cambia per l'utilizzo di A e B nei vari contesti? > covarianza-controvarianza

- Cosa vuol dire dal punto di vista concettuale e pratico (del design) il poter avere diverse visibilità? > pubblica-privata
  - Cosa tengo privato? Perchè?

- Quando un linguaggio è strongly typed e quando è weakly typed e perchè? [2]
  - in `C` alcune cose si "rompono" anche se lo definiamo strongly typed, cosa accade?
  - `Java` è strongly typed? perchè?
  - solo a livello statico o ci possono essere più parti? vedi tipaggio statico e dinamico
- Cos'è un tipo somma? Cos'è un tipo prodotto? [2]
  - un esempio quale potrebbe essere?
- Cos'è il duck typing e connessione con la structural typing? [2]
  - è riferito a qualcosa di specifico?
  - con le strutture invece?
  - dal punto di vista di espressibilità qual è la loro differenza?
  - il tipaggio strutturale è più o meno specifico?
  - a livello di espressività del linguaggio cosa cambia? > static typing
  - pro e contro > espressività maggiore => rischi > > dinamico vs statico
- Differenza tra la definizione intensionale ed estensionale nella definizione dei tipi
  - un esempio tipico di tipo estensionale > enumerators
  - che differenza hanno gli `int` dagli `enum`?
- Differenza tra `void` e `unit`? > `unit` rappresenta un insieme singoletto, restituendo un valore unico; mentre `void` rappresenta un insieme vuoto non restituendo nulla.
- ! Differenza tra sistemi basati su classi e sistemi basati su prototipi
  - dal punto di vista delle garanzie cosa possiamo dare nel caso di uno e nel caso dell'altro?
  - guardando a strongly typed e weakly typed, qual è la relazione?
  - quando un linguaggio è weakly typed?
  - un esempio di duck typing dove si riesce a spaccare le garanzie dei tipi del sistema come in `C`, con strongly typed.
    - `Java` è strongly typed, però…
    - qualcosa è da fare comunque in dinamico, si può fare comunque con strongly typed? > si può scollegare . . .
- ! Cosa vuol dire scope statico e dinamico, a cosa ci riferiamo?
  - come avviene il binding?
- ! Quali sono i principali valori fondanti dell'object orientation?
  - per avere un'implementazione quale polimorfismo è il più adatto?

- Differenza tra typed casting e type coertion [3]
  - un esempio
  - perchè non possiamo utilizzare questi costrutti all'interno del polimorfismo generale in `Java`?
- Cosa sono le eccezioni? A cosa servono?
  - se mettiamo `throw` e mettiamo la gestione interna?
  - come possiamo definire le nostre eccezioni, come si integrano in `Java`?
- Differenza tra polimorfismo ad hoc e dynamic dispatch, cosa sono?
  - cosa significa "overloading"?
  - dal punto di vista dell'implementazione, come differenziamo i due codici tra polimorfismo ad hoc e dynamic dispatch?
    - con che metodo si possono implementare?
- Differenza tra abstract data types e oggetti
  - dal punto di vista implementativo?
  - posso farli convivere nello stesso codice? perchè? > no perchè gli abstract data types impongono l'implementazione scelta all'inizio, invece gli oggetti possono essere sovrascritti ereditati etc …
- Differenza tra ereditarietà singola ed ereditarietà multipla
  - Alcuni metodi per risolvere le
- Come i tipi possono aiutare la programmazione?
- Cos'è il doc vector e qual è il suo utilizzo?
  - dove viene allocato il suo contenuto?
- Se le funzioni nei linguaggi fossero come le funzioni matematiche, come potremmo ottimizzarne la loro esecuzione?
  - come si scriverebbe?
  - un esempio > var globale lo chiamo una volta la incremento, la ritorno etc …
  - che ottimizzazioni possiamo fare con le funzioni matematiche? > di memoria: una tabella, per salvare i valori e lo riporto.
- Gestione degli errori nelle varie modalità (result types) e altre modalità
  - cosa abbiamo dal punto di vista pratico con il result type?
  - un'altra possibilità che possiamo chiedere al chiamante di darci per capire meglio cos'è? > il tipo `result`
- Cosa sono i dangling pointers e le dangling reference?
  - quando abbiamo i wide pointers/reference? (initialized pointers)