Consegna 5
Misconception

Lorenzo Dentis, lorenzo.dentis@edu.unito.it

28 marzo 2023

1 Domande

Partendo dall'appendice A della tesi Visual Program Simulation in Introductory Programming Education.

1. Provare a classificare le misconceptions in base al tipo di difficoltà (sintattica, concettuale, strategica), scrivetene alemno 5 per categoria nella consegna.
2. Data la vostra esperienza (di studenti delle superiori, di studenti universitari, alcuni di voi come tutor, alcuni di voi come persone che fanno ripetizioni), provate ad elencare "misconception" nella programmazione in cui vi siete imbattuti, personalmente o in altri.
3. Come le avete risolte?

2 Risposte

2.1 1 Classificare misconseptions

Misconception **sintattiche**:

1. **[10]**Variables always receive a particular default value upon creation.
2. **[38]**A return values does not need to be stored (even if one needs it later)
3. **[80]**An object is a subset of a class. / A class is a collection of objects.
4. **[125]**Cannot have methods with the same name in different classes
5. **[160]**Confusing textual representations with each other, e.g., the string "456" with the number.

Misconception **concettuali**:

1. **[8]**Magical parallelism: several lines of a (simple non- concurrent) program can be simultanenously active or known.
2. **[16]**Assignment moves a value from a variable to another.
3. **[47]**Subprograms can (routinely) use the variables of calling subprograms.

4. **[67]**Assigning to an object causes it to become equal to the assigned object.
5. **[124]**Objects 'know' which methods are operating on them (rather than method calls 'knowing' which object they operate on)

Misconception **strategiche**:

1. **[4]**The system does not allow unreasonable operations.
2. **[17]**The natural-language semantics of variable names affects which value gets assigned to which variabl
3. **[33]**loops terminate as soon as condition changes to false.
4. **[54]**Null model of recursion: recursion is impossible.
5. **[158]**Confusion between data in memory and data on screen.

2.2 2/3 Misconseption incontrate e come sono state risolte

11 Primitive assignment works in opposite direction. Probabilmente questa misconception deriva dalla matematica o dal metodo di scrittura "da sinistra a destra", però ricordo che quando ho iniziato a programmare trovavo più naturale che il valore venisse preso da sinistra e portato a destra dell'uguale. Questa misconception è stata di facile risoluzione in quanto una volta identificato cosa stavo sbagliando è stato solo una questione di ricordarmi "l'assegnamento va verso sinistra" o alla peggio lanciare il programma, realizzare il mio errore e girare l'assegnamento.

Una misconseption simile alla *22 Unassigned variables of primitive type (in Java) have no memory allocated.* ma diametralmente opposta mi è capitato di vederla nel corso di SO. Il collega sosteneva che si poteva allocare dinamicamente memoria come in java, senza bisogno di chiamate alla `malloc`.
Ad esempio tentava di fare la seguente operazione:

```
float read_and_process(int n)
{
    float vals[n];
    ....
}
```

Il motivo era che al posto di compilare da riga di comando con i parametri specificati dal professore (usando C90) il collega usava un IDE che di default aveva impostato C99. Per risolvere la misconception è bastato che il professore gli mostrasse il changelog di C99 in cui venivano introdotti i variable-length array e lo obbligasse a compilare da riga di comando.

Ma la misconseption più comune che mi è capitato di incontrare è stata la *1 The computer knows the intention of the program or of a piece of code, and acts accordingly.*Soprattutto nelle scuole superiori succedeva spessissimo che venissero giustificati errori nel codice con la frase "si ma io ovviamente intendevo questo" e la risposta del mio professore era sempre "il computer è una macchina stupida, si limita a fare ciò che gli dite di fare". Trovo che questa misconception

sia la più pericolosa ed anche la più difficile da "risolvere" in quanto spesso durante la programmazione bisogna fare degli assunti sul funzionamento di un programma, una libreria o un sistema ipotizzando che il sistema si comporti in un certo modo piuttosto che un altro. Lei professoressa citava ad esempio il "perdere il filo" dello sviluppo durante il passaggio da una fase all'altra dello sviluppo software in SAS, questa cosa capitava costantemente. Ricordo che nel mio gruppo ricontrollavamo più e più volte di aver considerato tutto e comunque ci si rendeva conto di aver perso pezzi tra una fase e l'altra e si doveva tornare alla fase precendente per aggiustare. Credo che questa misconception si risolva in due soli modi: tramite l'esperienza con l'uso di un sistema specifico/la ripetizione di una procedura (nel caso dello sviluppo software) e tramite la lettura della documentazione dove c'è scritto "il sistema fa questo e non fa quest'altro", operazione che richiede sempre un certo dispendio di tempo.