

Lösningsförslag till omtentamen för TDA540

Objektorienterad Programmering

Institutionen för Datavetenskap

CTH HT-16, TDA540

Dag: 2017-04-12, *Tid:* 14.00-18.00

Uppgift 1

- a) En konstruktör är en del av en klass och skapar objekt. En konstruktör brukar initialisera instansvariabler.
- b) En klass kan ärva metoder och variabler från en annan (super) klass. Arv är ett sätt att abstrahera och återanvända kod.
- c) Med en tilldelning (i en sats) uppdaterar vi en variabels värde.
- d) En kompilator är ett verktyg som översätter ett program (skrivt i källkod i ett visst programspråk) till maskinkod som kan köra på en dator. Dvs en kompilator transformerar källkod till ett exekverbart program.

Uppgift 2

- 4 Metoden `f` tar ett heltal (`int`) som argument och kan inte appliceras på ett reeltal (`double`).
- 6 Metoden `h` har två parametrar och kan inte bli anropad med tre.
- 8 Det finns ingen metod `b`

Uppgift 3

```
import java.util.ArrayList;
import java.util.List;
import java.util.NoSuchElementException;

public class Tokenizer {
    private char[] input;
    private String delimiters;
    private int i = 0;           // position of the 'cursor'

    public Tokenizer(String input, String delimiters) {
        this.input = input.toCharArray();
        this.delimiters = delimiters;
        consumeDelimiters();     // Make sure we can read first token
    }

    public String nextToken() {
        String token = "";

        while (hasMoreTokens()) // Read until we hit a delimiter
            token += input[i++];

        consumeDelimiters();     // Consume all delimiters

        if (token.equals(""))
            throw new NoSuchElementException();
        else
            return token;
    }

    public List<String> allTokens() {
        List<String> tokens = new ArrayList<>();

        while (hasMoreTokens())
            tokens.add(nextToken());

        return tokens;
    }

    public boolean hasMoreTokens() {
        return i < input.length && delimiters.indexOf(input[i]) < 0;
    }

    private void consumeDelimiters() {
        while (i < input.length && delimiters.indexOf(input[i]) >= 0)
            i++;
    }
}
```

Uppgift 4

(3 poäng)

a) En statisk typ är den deklarerade typen av en variabel. Till exempel:

```
int i;  
Tokenizer t = new Tokenizer("Alex Gerdes", " ");
```

då är `int` statiska typen av `i` och `Tokenizer` av `t`.

b) En dynamisk typ är typen av själva objektet som en variabel pekar på. Till exempel:

```
Object o = t;
```

Objektet `o` med *statiska* typen `Object` är en referens till ett objekt med typen `Tokenizer`.
Dvs att *dynamiska* typen av `o` är `Tokenizer`.

c) Med typkonvertering kan vi vandla om statiska typen av ett objekt. Till exempel:

```
Tokenizer s = (Tokenizer)o;
```

Här typkonverterar vi statiska typen av `o` till `Tokenizer`.

Uppgift 5

Utskriften blir:

```
Output is "a = 2 b = 1"
```

- Antal variabler är 10: 2 in `Wrapper`, 3 in `Swapper`, 5 in `run()`
- Antal objekt är 6: 2 `Integer`, 2 `Wrapper`, 1 `Swapper`, 1 `String`

Uppgift 6

- a)
- ```
static boolean isPrime(int n) {
 for (int k = 2; k < n; k++)
 if (n % k == 0)
 return false;
 return true;
}
```
- b)
- ```
static boolean isComposite(int n) {  
    return !isPrime(n);  
}
```

c)

```

static int[] primes(int n) {
    int[] buffer = new int[n];
    int p = 0;
    for (int k = 2; k <= n; k++)
        if (isPrime(k)) {
            buffer[p] = k;
            p += 1;
        }
    int[] result = new int[p];
    for (int k = 0; k < p; k++)
        result[k] = buffer[k];
    return result;
}

```

Uppgift 7

- a)
1. 4
 2. 4
 3. unknown
 4. unknown
 5. Made in Göteborg
- b)
1. error: interfaces cannot be instantiated
 2. accepted: subtype to supertype (a Car is a Vehicle)
 3. accepted: subtype to supertype (a Volvo is a Car)
 4. error: a supertype cannot be converted to a supertype (a Car may not be a Volvo)
 5. accepted: subtype to supertype (a Volvo is a four wheeler)
 6. error: abstract classes cannot be instantiated
 7. accepted: subtype to supertype (a Car is a Wheeler)
 8. accepted: same type
 9. accepted: same type
 10. error: a list of cars is not a subtype of a list of vehicles (even if a Car is a Vehicle)

Uppgift 8

a)

```

public Integer get(int index) {
    if (0 <= index && index < size)
        return data[index];
    else
        return null;
}

```

- b)

```
public Integer remove() {  
    if (size > 0) {  
        size -= 1;  
        return data[size];  
    } else  
        return null;  
}
```
- c)

```
public void add(Integer item) {  
    if (size < 10_000) {  
        data[size] = item;  
        size += 1;  
    } else  
        throw new FullException();  
}
```