

## Lösningförslag: Instuderingsfrågor, del C

### Uppgift 1.

Top-down-design innebär att man betraktar det ursprungliga problemet på en hög abstraktionsnivå och bryter ner det ursprungliga problemet i ett antal delproblem. Var och en av dessa delproblem betraktas sedan på en lägre abstraktionsnivå. Om nödvändigt bryts delproblemen ner i mindre och mer detaljerade delproblem. Denna process upprepas till man har delproblem som är triviala att lösa.

Fördelen med top-down-design är att man lösa det delproblemen i det ursprungliga problemet steg för steg istället för att direkt göra en fullständig lösning.

### Uppgift 2.

Resultattypen **void** betyder att metoden inte returnerar något värde.

### Uppgift 3.

De formella parametrarna definieras i metodens parameterlista och är lokala inom metoden. De formella parametrarna får sina värden när metoden anropas via de parametrar som anges vid anropet. De parametrar som anges vid anropet kallas för aktuella parametrar. När en metod anropas sker en bindning mellan de aktuella och formella parametrarna. Detta sker genom att den formella parametrarna tilldelas värdet av motsvarande aktuell parameter. Således gäller vid anrop till en metod att de de aktuella parametrarna måste överensstämma med de formella parametrarna i antal, typ och ordning.

### Uppgift 4.

- a) Falskt.
- b) Falskt. En **void**-metod returnerar inget värde och behöver således inte någon **return**-sats
- c) Falskt.
- d) Falskt.
- e) Sant.
- f) Falskt.

### Uppgift 5.

- a) Kompileringsfel! Metoden **methodOne** är en **void**-metod. Det går inte att tilldela variabeln **d** returvärdet från metoden, eftersom det inte finns något returvärde.
- b) Korrekt. Den formella parametrarna **one** till metoden har typen **double** och den aktuella parametrarna **a** har typen **int**, men **int** är typkompatibel med **double**.
- c) Kompileringsfel! Den aktuella parametrarna **c** av typen **double** är inte typkompatibel med den formella parametrarna **one** av typen **int**.
- d) Korrekt. Den formella parametrarna **two** till metoden har typen **double** och den aktuella parametrarna **b** har typen **int**, men **int** är typkompatibel med **double**.
- d) Korrekt. Metoden returnerar ett värde av **int** som tilldelas variabeln **d** av typen **double**, men **int** är typkompatibel med **double**.
- e) Kompileringsfel! Antalet aktuella parametrarna överensstämmer inte med antalet formella parametrarna.

### Uppgift 6.

48

### Uppgift 7.

103

### Uppgift 8.

- a) Metoden är deklarerad som **void**, dvs att metoden inte skall returnera något värde, dock returneras en **int**! En korrekt metod har följande utseende:

```
public int methodOne (int one) {  
    return one + one;  
} //methodOne
```

- b) Metoden skall returnera en **boolean**, men om villkoret är falskt returneras inget värde!. En korrekt metod har följande utseende:

```
public boolean methodTwo (int a, int b) {  
    if (a > 2*b)  
        return true;  
    else  
        return false;  
} //methodTwo
```

### Uppgift 9.

Metoden returnerar **true** om parametern x är ett jämt heltal och **false** om parametern x är ett udda heltal. Ett lämpligare namn är `isEven`.

### Uppgift 10.

Metoden beräknar  $n^k$ . Ett lämpligare namn på metoden är `power`.

### Uppgift 11.

6

### Uppgift 12.

- a) 

```
public static int lastDigit(int number) {  
    return Math.abs(number % 10);  
} //lastDigit
```
- b) 

```
public static int firstDigit(int number) {  
    int temp = Math.abs(number);  
    while (temp > 10) {  
        temp = temp / 10;  
    }  
    return temp;  
} //firstDigit
```
- c) 

```
public static int nrOfDigits(int number) {  
    int temp = Math.abs(number);  
    int nrDigits = 1;  
    while (temp > 10) {  
        nrDigits = nrDigits + 1;  
        temp = temp / 10;  
    }  
    return nrDigits;  
} //nrOfDigits
```
- d) 

```
public static int largestDigit(int number){  
    int temp = Math.abs(number);  
    int max = 0;  
    while (temp > 0) {  
        int digit = temp % 10;  
        if (digit > max)  
            max = digit;  
        temp = temp / 10;  
    }  
    return max;  
} // largestDigit
```

### Uppgift 13.

- a) **public static int** squareSum(**int** from, **int** to) {  
    **int** sum = 0;  
    **for** (**int** i = from; i <= to; i = i + 1) {  
        sum = sum + i\*i;  
    }  
    **return** sum;  
} //squareSum
- b) **public static double** powerSum(**double** x, **int** n) {  
    **double** term = x;  
    **double** sum = 1;  
    **for** (**int** i = 1; i <= n; i = i + 1) {  
        sum = sum + term;  
        term = term \* x;  
    }  
    **return** sum;  
} //powerSum