Programmazione Funzionale

Esercitazione 2 – Pattern Matching

Esercizio 1. Indicare il valore restituito dai seguenti pattern matching in OCAML, e indicare quale dei seguenti pattern matching invece sollevera un eccezione:

- `match x with 0 -> 0.`
- `match 1 with 0-> 0.`
- `match 0 with 0-> 0.`
- `match 0 with 0 -> (match 0 with 0 -> 0).`
- `match (2,2) with x -> 1.`

Esercizio 2. Indicare il valore o l'eccezione restituito dai seguenti pattern matching in OCAML, dopo aver dichiarato

```
let f = function x -> x+1;;
```

- `match Red with Red 1 -> Blue.`
- `match Red with Blue -> Red.`
- `match Red with Red -> Red.`
- `match f(1) with Red 1 -> 0.`

Esercizio 3. Indicare il valore o l'eccezione restituito dai seguenti pattern matching in OCAML, dopo aver dichiarato

```
let f = function x -> x+1;;
type color = Red | Blue | Green;;
```

- `match f(1) with Red 2 -> 0.`
- `match f(1) with Red f(1) -> 0.`
- `match (Red,Blue) with (x,y) -> (x,x) | (Red,_) -> (Green,Green).`

Esercizio 4. Indicare se i pattern matching sono esaustivi e il loro valore. Se i matching non sono esaustivi completarli (solo aggiungendo casi) in modo che il pattern matching diventi esaustivo. Quando non e possibile completare il pattern matching spiegare perche.

Assumiamo di aver dichiarato

```
type somma = I of int | J of int
type intero = Zero | Succ of intero
```

- `match I 3 with I 3 -> 0.`
- `match I 3 with I x -> 0.`
- `match I 3 with J x -> 0.`
- `match I 3 with I x -> 0 | J x -> 1.`
- `match Succ(Zero) with I x -> 0.`
- `match Succ(Zero) with x -> 0.`
- `match Succ(Zero) with Succ(x)-> 0.`

Esercizio 5. Mediante pattern matching definire una funzione di tipo $int \times int \to int$ che applicata a una coppia di interi (n, m) restituisce

- `true` per qualsiasi coppia $(0, m)$ quando $m \neq 0$.
- `true` per qualsiasi coppia (n, m) quando $m \in \{3, 4\}$.
- `true` per la coppia $(1, 0)$.
- `false` in qualsiasi altro caso.