1. There are four violations:

1. The `then` keyword is not used in Scala (static error).
2. Local variable `s` must be initialized as part of its definition (static error).
3. The first branch of the `if` evaluates to `Unit`, which does not match `Int`, the type of the second branch and the declared return type of the function (static error).
4. Since `j = -1`, the reference `a(j)` will be out of bounds (checked runtime error).

Note that Scala has no unchecked runtime errors.

2. (a) `int`
   (b) `(a → (a → a))`
   (c) `((bool → a) → a)`
   (d) Not typable: `x` cannot be both a function and an integer.
   (e) `((a → bool) → (a → bool))`

3. Since either arm of the `catch` might yield the overall value, the arms must have the same type, but are otherwise unconstrained (just as for an `if` expression). Since `throw` never actually yields a value, it can safely be assigned any type whatever, and we need that flexibility in order to use it in arbitrary positions in the code.

\[
\begin{align*}
TE ⊢ (\text{throw}) : t \\
TE ⊢ e_1 : t & \quad TE ⊢ e_2 : t \\
\overline{TE} ⊢ \text{(catch } e_1 e_2) : t
\end{align*}
\]

4. (a)

   OO programmer hacks classes
   Functional programmer uses pattern matching

   (b)

   Scala programmer hacks code
   Scala programmer hacks code

5.

```scala
case class P(i: Int, u: T, v: T) extends T {
  def f() = i * u.f() + v.f()
}

case class Q(b: Boolean) extends T {
  def f() = if (b) 1 else 0
}
```
6.a. f: x, z. g: y.

(b)

def M2(x:Boolean,y:Int,z:Int) =
   R(w => if (x) z + w else w - 42,
       w => w + y)

7. Under method A, \(s ++ "x"\) and "x" ++ \(s\) will both take time proportional to \(|s|\), because the entire string must be copied. Under method B, \(s ++ "x"\) will still take time proportional to \(|s|\), because the string must be traversed, but "x" ++ \(s\) will take only unit time. So comparing the execution times of the following programs should do the trick: if program 1 runs much faster than program 2, method B is being used; if the runtimes are about the same, it's method A.

Program 1:

\[
\begin{align*}
\text{s} &= ""
\text{for i = 1 to 1000000 do}
\text{s} &= "x" ++ \text{s};
\end{align*}
\]

Program 2:

\[
\begin{align*}
\text{s} &= ""
\text{for i = 1 to 1000000 do}
\text{s} &= \text{s ++ "x"};
\end{align*}
\]

8.

def count(b:B,x:A) : Int = b match {
   case EmptyB => 0
   case InsertB(b,y) => count(b,x) + (if (x == y) 1 else 0)
   case DeleteB(b,y) => (count(b,x) - (if (x == y) 1 else 0)) max 0
   case UnionB(b1,b2) => count(b1,x) + count(b2,x)
}