

1. 下列模式能否与类型为 `int list` 的 `L` 匹配成功？如果匹配不成功，指出该模式的类型？（假设 `x` 为 `int` 类型）

`x::L` 成功

`_::_` 成功

`x::(y::L)` 当 `list` 成员大于等于 2 时，成功

`(x::y)::L` 错误，匹配的是 `int list list`

`[x, y]` 当 `list` 成员等于 2 时，成功

2. 试写出与下列表述相对应的模式。如果没有模式与其对应，试说明原因

`list of length 3`: `[x,y,z]`

`lists of length 2 or 3`: 没有，可以拆分成 `[x,y]` 和 `[x,y,z]` 两个模式分别进行描述

`Non-empty lists of pairs`: `(x,y)::L`

`Pairs with both components being non-empty lists`: `(x::L1, y::L2)`

3. 分析下述程序段（左边括号内为标注的行号）

第 4 行的 `x`: `int`, 2

第 5 行的 `m`: `real`, 12.4

第 6 行的 `x`: `int`, 9001

第 14 行的计算结果: 27

4 指出下列代码的

```
(* f : int -> int *)
```

```
fun f (3 : int) : int = 9
```

```
| f_ = 4
```

```
(*circ : real -> real *)
```

```
fun circ (r : real) : real = 2 2.0 * pi * r
```

```
(* semicirc : real -> real *)
```

```
fun semicirc : real = pie pi * r
```

```
(*area : real -> real *)
```

```
fun area (r : int real) : real = pi * r * r
```

5. 在提示符下依次输入下列语句，观察并分析每次语句的执行结果

```
3 + 4;      (* val it=7 *)
```

3 + 2.0; (\* 错, real 和 int 类型不匹配 \*)  
it + 6; (\* val it = 13 \*)  
val it = "hello"; (\* val it = "hello" \*)  
it + "world"; (\* 错, 要用^操作符拼接 \*)  
it + 5; (\* 错误, string 和 int 类型不匹配 \*)  
val a = 5; (\* val a = 5 \*)  
a = 6; (\* val it = false \*)  
a + 8; (\* val a = 13 \*)  
val twice = (fn x => 2 \* x); (\* val twice = fn : int -> int \*)  
twice a; (\* val it = 10 \*)  
let x = 1 in x end; (\* 错, 给 x 赋值要用 val x = 1 \*)  
foo; (\* 错误, foo 未绑定 \*)  
[1,"foo"]; (\* 错, list 里不能包含不同类型成员 \*)