```
#Lab 5 (Fri) solutions
#Suprakash Datta

#Q1:
q1 := proc(a :: integer, b :: integer, c :: integer)
 if a = b and a ≠ c then return 1; #enumerate all possible orders of 3 elements
 elif a = c and a ≠ b then return 1;
 elif b = c and b ≠ a then return 1;
 else return 0;
 end if;
end proc;
```

```
proc(a::integer, b::integer, c::integer)                                    (1)
    if a = b and a <> c then
        return 1
    elif a = c and a <> b then
        return 1
    elif b = c and b <> a then
        return 1
    else
        return 0
    end if
end proc
```

```
q1(2, 1, 1); q1(1, 2, 1); q1(1, 1, 2); q1(1, 2, 3);
                                        1
                                        1
                                        1
                                        0                                   (2)
q1(1, 1, 1);
                                        0                                   (3)
#Q2

numdig := n→1 + trunc(log10(n));#get the number of digits of an integer
                                n→1 + trunc(log10(n))                       (4)
numdig(102);
                                        3                                   (5)
ithdig := proc(n :: integer, i :: integer)#get the ith digit(from the right) of an integer
 return iquo(irem(n, 10^i), 10^(i - 1));
 end proc;
        proc(n::integer, i::integer) return iquo(irem(n, 10^i), 10^(i - 1)) end proc   (6)
 digset := proc(n :: integer)
local m;
 m := abs(n);#takes care of negative integers
{seq(ithdig(m, i), i = 1 ..numdig(m))};
 end proc;
```

```
proc(n::integer) local m; m := abs(n); {seq(ithdig(m, i), i = 1 ..numdig(m))} end proc
```
(7)

```
digset(-434); digset(433);
```

{3, 4}

{3, 4} **(8)**

```
#Q3 Note the extra check if all digits are used. You will not be penalized if you did not do this.
q4 := n→piecewise({0, 1, 2, 3, 4, 5, 6, 7, 8, 9}\ digset(n) ≠ { }, min({0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
    \ digset(n)), {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}\ digset(n) = { }, "All digits used");
```

```
n→piecewise({0, 1, 2, 3, 4, 5, 6, 7, 8, 9}\digset(n) ≠ { }, min({0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
    \digset(n)), {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}\digset(n) = { }, "All digits used")
```
(9)

```
q4(23456789); q4(1234567890);
```

0

"All digits used" **(10)**

```
#Q4
NotPowerOf3or2 :=proc(n :: integer)
 local set1, set2;
 set1 := {seq(i, i = 1 ..n)} minus {seq(3^i, i = 0 ..n)};
    #could have restricted the range to 1 +trunc(log3(n))
 set2 := set1 minus {seq(2^i, i = 0 ..n)};
return set2;
end proc;
```

```
proc(n::integer)
    local set1, set2;
    set1 := minus({seq(i, i = 1 ..n)}, {seq(3^i, i = 0 ..n)});
    set2 := minus(set1, {seq(2^i, i = 0 ..n)});
    return set2
end proc
```
(11)

```
NotPowerOf3or2(20);
```

{5, 6, 7, 10, 11, 12, 13, 14, 15, 17, 18, 19, 20} **(12)**

```
NotPowerOf3or2(10);
```

{5, 6, 7, 10} **(13)**

```
#Q5 -- make sure you return 1 even if all the digits are 1 or 2
All12 := n→piecewise(digset(n) minus {1, 2} = { }, 1, digset(n) minus {1, 2} ≠ { }, 0);
```

```
n→piecewise(digset(n)\{1, 2} = { }, 1, digset(n)\{1, 2} ≠ { }, 0)
```
(14)

```
All12(21112);
```

1 **(15)**

```
All12(2111123);
```

0 **(16)**

```
All12(222);
```

1 **(17)**

```
All12(111);
```

1 **(18)**