```
>
>
>
> # Lab 9 (Tue) solutions
>
>
> #Q1
>
> rprint :=proc(n :: integer, num :: integer)
   if num = 0 then return;
   else print(n);
   rprint(n, num − 1);
   end if;
   end proc
```

```
rprint := proc(n::integer, num::integer)                                    (1)
    if num = 0 then return else print(n); rprint(n, num − 1) end if
end proc
```

```
>
> rprint(3, 3);
                                        3
                                        3
                                        3                                   (2)
> #Q2
>
> q2 :=proc(n :: integer)
   if n = 1 then print(1);return;
   else
   q2(n − 1);
   rprint(n, n);
   end if;
   end proc;
```

```
q2 := proc(n::integer)                                                      (3)
    if n = 1 then print(1); return else q2(n − 1); rprint(n, n) end if
end proc
```

```
> q2(4);
                                        1
                                        2
                                        2
                                        3
                                        3
                                        3
                                        4
                                        4
                                        4
                                        4                                   (4)
```

```
>
> #Q3
  notprime :=proc(L :: list, n :: posint)#find the sum of non-prmes in L[1..n]
   if n = 1 and isprime(L[n]) = true then return 0;
   elif n = 1 and isprime(L[n]) = false then return L[n];
   elif isprime(L[n]) = true then return notprime(L, n − 1);
   else
  return L[n] + notprime(L, n − 1);
   end if;
   end proc;
```

```
notprime := proc(L::list, n::posint)                                    (5)
    if n = 1 and isprime(L[n]) = true then
        return 0
    elif n = 1 and isprime(L[n]) = false then
        return L[n]
    elif isprime(L[n]) = true then
        return notprime(L, n − 1)
    else
        return L[n] + notprime(L, n − 1)
    end if
end proc
```

```
>
>
>
> L := [11, 21, 31, 42];
```

$$L := [11, 21, 31, 42] \quad (6)$$

```
> notprime(L, 4);
```

$$63 \quad (7)$$

```
>
> #Q4
>
> q4proc :=proc(L :: list)
   local LL :
   if frac( nops(L) / 2 ) ≠ 0 then print("Error: odd length list"); return 0;
   elif nops(L) = 2 then return L[1]·L[2];
   else
   LL := L[3 ..nops(L) ];
   return L[1]·L[2] + q4proc(LL);
   end if;
   end proc;
```

```
q4proc := proc(L::list)                                                 (8)
    local LL;
    if frac(1/2*nops(L)) <> 0 then
        print("Error: odd length list"); return 0
    elif nops(L) = 2 then
```

```
        return L[1]*L[2]
    else
        LL := L[3..nops(L)]; return L[1]*L[2]+q4proc(LL)
    end if
end proc
>
> q4proc([6, 5, 1, 2, 3, 5]);
                                47                                (9)
>
>
```