Exercise 1

We consider adding the two following expressions to the While language.

- \( b \ ? \ e_1 : e_2 \) is a 'conditional' arithmetic expression. Its value is that of \( e_1 \) if \( b \) evaluates to true, and that of \( e_2 \) otherwise.
- \( \text{for } x \text{ in } e_1 .. e_2 \text{ do } S \), the for statement (we suppose \( x \) is declared before we encounter this statement).

Give the corresponding code generation functions.

Exercise 2

We consider the following piece of code:

```plaintext
proc p() is
  begin
  var z;

  proc p1() is
    begin
    proc p2(x,y) is z:=x+y;
    z:=0;
    call p2(z+1,3);
    end;

  proc p3(x) is
    begin
    var z;
    call p1();
    z:=z+x;
    end;
    call p3(42);
  end;
```

1. Draw the execution stack at the moment procedure \( p2 \) is called.
2. Give the code generated for procedure \( p2 \).
3. Give the code generated for procedure \( p1 \).
Exercise 3

We consider the following piece of code:

```plaintext
proc p() is
  begin
  var x;

  proc p1(a) is
    begin
    var x1;

    proc p2(b, c) is
      begin
      var x2;
      x2 := c;
      x := x1 + x2 + b; (*)
      end;

      x1 := a;
      call p2(x + 1, a); (**) 
      x := 2;
      end;

    x := 0;
    p1(5);
  end;
end;
```

1. Draw the content of the execution stack when \texttt{p2} is called.
2. Give the code generated for line (**).
3. Give the code generated by line (*).