Taylan Unal

2. (3 points) Code the Ackermann function in DrRacket, which is defined as follows:

$$A(m,n) = \begin{cases} n+1 & \text{if } m = 0 \\ A(m-1,1) & \text{if } m > 0 \text{ and } n = 0 \\ A(m-1, A(m, n-1)) & \text{if } m > 0 \text{ and } n > 0 \end{cases}$$

To check your result, use DrRacket to compute the result of $A(3,5)$, which should be 253. You only need to submit your Scheme code for the Ackermann function.

3. (3 points) John McCarthy is a famous computer scientist who designed LISP. He once proposed a function called the McCarthy 91 function, defined as follows:

$$\mathrm{Mac}(n) = \begin{cases} n - 10 & \text{if } n > 100 \\ \mathrm{Mac}(\mathrm{Mac}(n+11)) & \text{if } n \leq 100 \end{cases}$$

Write this function in Scheme. Try calling the function with a few numbers less than 100 and see what the results are.

4. (3 points) Some credit-card companies pay back a small portion of the charges a customer makes over a year. One company returns

   (a) 0.5% for the first $1000 of charges,
   (b) 0.75% for the next $1000 (that is, the portion between $1000 and $2000),
   (c) 1.0% for the next $1500 (that is, the portion between $2000 and $3500),
   (d) and 1.5% for everything above $3500.

   Thus, a customer who charges $400 a year receives $2, which is 0.5% * 400, and one who charges $1,400 a year receives $8, which is 5 = 0.5% * 1000 for the first $1000 and 0.75% * 400 = 3 for the next $400.

   Define the function `payback`, which consumes a charge amount and computes the corresponding pay-back amount.