Common Lisp

You are asked by your manager to write a Lisp macro, \texttt{itercall}. Evaluating \texttt{(itercall \ F \ E)} evaluates \(E\), which is expected to yield a non-negative integer \(n\). It then executes the function calls \((F \ 1), \ldots, (F \ n)\) in succession, and returns \texttt{nil}.

(a) Your first version of the macro expands to a loop, which uses the symbol \(i\) as an index variable and the symbol \(n\) to store the initial value of \(E\). Present the code for this version. [5 marks]

(b) Your manager complains that the function

\begin{verbatim}
(defun test1 (i) (itercall (lambda (x) (print (cons x i))) 10))
\end{verbatim}

does not work as expected. Explain the problem and suggest how to fix it by modifying the macro. [4 marks]

(c) Your manager requests a final modification: \texttt{(itercall \ F \ E)} should generate straight-line code instead of a loop provided \(E\) is an integer constant less than twenty. Present the code for this version. Will it run faster than the previous versions? [11 marks]

Note: \texttt{(integerp \ x)} tests whether \(x\) is an integer. Each time \texttt{(gensym)} is called, it returns a new symbol not previously used in the Lisp system.