**Problem 1 (15 points)**

Consider the following recursive algorithm that on input two integers \( x, y \) outputs some other integer \( f(x, y) \).

\[
prog(x, y) = \begin{cases} 
  \text{if } (y < 0) \text{ then } prog(-x, -y) \\
  \text{else if } (y == 0) \text{ then } 0 \\
  \text{else if } \text{even}(y) \text{ then } prog(2 \cdot x, y/2) \\
  \text{else } (x + prog(x, y - 1)) 
\end{cases}
\]

Notice, the program computes \( y/2 \) only when \( y \) is even, so the division is always exact.

(a) Compute \( prog(x, y) \) on a few test values of your choice, and determine the function \( f(x, y) \) computed by the program.

Your answer should include at least 5 tests (e.g., \( prog(4, 0) = 0 \)), but you do not need to include the intermediate steps of the computation. (You may run the program on a computer if you like.) The function \( f(x, y) \) should be specified as a simple arithmetic expression in the variables \( x \) and \( y \).

(b) For this part, assume \( y \geq 0 \). Prove, by induction on \( y \), that \( prog(x, y) \) correctly computes \( f(x, y) \). Your answer should start with a formal statement of the property you intend to prove (with properly quantified variables), followed by a clear and well written proof of the statement.

(c) Prove that \( prog(x, y) \) correctly computes \( f(x, y) \) for any integer inputs \( x, y \). (You may use the result of part (b) if you like.)

**Problem 2 (10 points)**

Consider the program in problem 1, and let \( t(x, y) \) be the running time of \( prog(x, y) \), measured as the total number of (recursive) calls made to \( prog \). For example, \( t(4, 0) = 1 \), just to account for the initial call to \( prog(4, 0) \), because \( prog(4, 0) = 0 \) does not make any recursive calls.

Determine an upper bound on \( t(x, y) \), and prove your answer correct. You do not need to determine the exact value of \( t(x, y) \). Any upper bound \( t(x, y) \leq ... \) is acceptable, but the closer the better.

**Problem 3 (10 points)**

Prove, by induction on \( n \), that for all \( n > 1 \),

\[
\sum_{k=1}^{n} \frac{1}{k^2} < 2 - \frac{1}{n}.
\]
Problem 4 (15 points)

Consider the following program, which takes as input an integer $n \geq 0$, and returns some other integer as a result.

\[
P(n) \{
    i := 0 ;
    s := 0 ;
    \text{while } (i < n) \text{ do}
    \quad i := i + 1
    \quad s := s + i
    \text{ return } s
\}
\]

(a) Compute $P(n)$ on a few test values of your choice, and determine the function $g(n)$ computed by the program.

Your answer should include at least 5 tests, but you do not need to include the intermediate steps of the computation. (You may run the program on a computer if you like.)

(b) Prove that the program $P(n)$ computes the function $g(n)$ by presenting an appropriate loop invariant for the program.

Your solution should include a clear description of the loop invariant, a proof that the loop invariant is correct, and a proof that the correctness of the program follows from the loop invariant.

Problem 5 (10 points)

Use the relation $(n+1)^2 = n^2 + 2n + 1$ to give a recursive program that, on input any integer $n \geq 0$, outputs $n^2$. (Your program is only allowed addition operations. No multiplications! You can compute $2 \cdot n$ as $n + n$.

(You can express your program using pseudo code, using a syntax similar to the one used in Problem 1.)

Prove the correctness of your program using mathematical induction.

Problem 6 (10 points)

Prove, using strong induction, that any group of $n \geq 8$ people can be divided into teams of size 3 or 5.