In this problem you have to implement several functions in Haskell. You do not need to ask permission to write auxiliary functions, of course you can!

1. Write a function \texttt{absValue :: Int \rightarrow Int} that, given an integer, returns its absolute value.

2. Write a function \texttt{power :: Int \rightarrow Int \rightarrow Int} that, given an integer \texttt{x} and a natural \texttt{p}, returns the \texttt{p}-th power of \texttt{x}, that is, \texttt{x^p}.

3. Write a function \texttt{isPrime :: Int \rightarrow Bool} that, given a natural, tells whether it is a prime number or not.

4. Write a function \texttt{slowFib :: Int \rightarrow Int} that returns the \texttt{n}-th element of the Fibonacci sequence using the recursive algorithm that defines it \((f(0) = 0, f(1) = 1, f(n) = f(n-1) + f(n-2) \text{ for } n \geq 2)\).

5. Write a function \texttt{quickFib :: Int \rightarrow Int} that returns the \texttt{n}-th element of the Fibonacci sequence using a more efficient algorithm.

\section*{Scoring}

Each function scores 20 points.

\begin{description}
\item [Sample input] \texttt{absValue (-666)}
\item [Sample output] \texttt{666}
\end{description}