Comparative Programming Languages

(a) In order to remove the overhead of a function call, a programmer decides to replace all calls to a function \( f \) with the macro \( F \), where \( f \) and \( F \) are defined as follows:

\[
\text{int } f(\text{int } x) \{ \text{return } x+x; \}
\]

\[
\#define F(X) (X)+(X)
\]

(i) Give two valid C expressions involving \( f \) which produce different results when \( F \) is substituted for \( f \). Justify your answer. [4 marks]

(ii) State the C language feature which can be used to correctly remove the overhead of a function call. [1 mark]

(b) Consider the following:

\[
\text{static struct link } \{
\text{int } v;
\text{struct link } *\text{next};
\} \text{ } *\text{head}=0;
\]

\[
\text{void convert(int } a[], \text{int len);}
\]

Write a function definition for convert which updates head to point to a linked-list containing the elements of \( a \) in the same order. You may assume \( \text{len} \) contains the number of elements in \( a \). [5 marks]

(c) Consider the following C++ declaration:

\[
\text{template<int } n> \text{int } \text{SumSquares}();
\]

(i) Using function specialisation, provide an implementation of \( \text{SumSquares} \) so that, given an integer \( N \), \( \text{SumSquares}<\text{N}>() \) returns:

\[
\sum_{i=1}^{N} i^2
\]

[5 marks]

(ii) Compare and contrast the functionality of the C preprocessor and the C++ template system. Explain why it is not possible to write a C preprocessor macro to implement \( \text{SumSquares} \). [5 marks]