Question and Answers Document

1. What are error recovery techniques in top down parsing?  
  
Top-down parsing is a parsing technique that starts with the goal symbol and tries to match it with the input string one symbol at a time. If a mismatch occurs, the parser can try to recover by backtracking and trying a different rule.  
  
There are several error recovery techniques that can be used in top-down parsing:  
  
\* \*\*Error anticipation:\*\* This technique involves trying to predict the next symbol in the input string and then backtracking if the prediction is wrong.  
\* \*\*Error guessing:\*\* This technique involves guessing the next symbol in the input string and then backtracking if the guess is wrong.  
\* \*\*Error recovery by repair:\*\* This technique involves repairing the input string so that it can be parsed correctly.  
  
2. What is Boot ‘Strapping?  
  
Boot ‘strapping is the process of initializing a computer system so that it can start running programs. This is usually done by loading a small program called a bootstrap loader into memory. The bootstrap loader then loads the operating system into memory, which allows the computer to start running programs.  
  
3. Difference between Top Down parsing and Bottom up parsing.  
  
Top-down parsing is a parsing technique that starts with the goal symbol and tries to match it with the input string one symbol at a time. Bottom-up parsing is a parsing technique that starts with the input string and tries to match it with the goal symbol one symbol at a time.  
  
The main difference between top-down parsing and bottom-up parsing is that top-down parsing is a predictive parser, while bottom-up parsing is a non-predictive parser. In top-down parsing, the parser tries to predict the next symbol in the input string and then backtracks if the prediction is wrong. In bottom-up parsing, the parser does not try to predict the next symbol in the input string. Instead, it tries to match the input string with the goal symbol one symbol at a time.  
  
4. What are ambiguous grammars? Give an example.  
  
An ambiguous grammar is a grammar that can generate more than one parse tree for a given input string. For example, the following grammar is ambiguous:  
  
```  
S -> NP VP  
NP -> Det N  
VP -> V NP | V  
Det -> "the" | "a"  
N -> "man" | "woman"  
V -> "saw" | "ate"  
```  
  
The input string "the man saw the woman" can be parsed in two different ways:  
  
\* S -> NP VP  
NP -> Det N  
Det -> "the"  
N -> "man"  
VP -> V NP  
V -> "saw"  
NP -> Det N  
Det -> "the"  
N -> "woman"  
  
\* S -> NP VP  
NP -> Det N  
Det -> "the"  
N -> "man"  
VP -> V  
V -> "saw"  
NP -> Det N  
Det -> "the"  
N -> "woman"  
  
5. What is Yacc? Explain the Syntax.  
  
Yacc is a parser generator that can be used to generate parsers for context-free grammars. Yacc is a recursive descent parser generator, which means that it generates parsers that work by recursively descending into the parse tree.  
  
The syntax of Yacc grammars is as follows:  
  
```  
%token <token-name>  
%left <production-name>  
%right <production-name>  
%nonassoc <production-name>  
  
<start-symbol> : <production>  
<production> : <non-terminal> | <non-terminal> <production>  
<non-terminal> : <terminal> | <non-terminal> <terminal>  
<terminal> : <character> | <string>  
```  
  
The `%token` directive defines a token. The `%left` and `%right` directives define the associativity of a production. The `%nonassoc` directive defines a production as non-associative.  
  
The `<start-symbol>` is the name of the start symbol of the grammar. The `<production>` is a production in the grammar. The `<non-terminal>` is a non-terminal symbol in the grammar. The `<terminal>` is a terminal symbol in the grammar.  
  
6. Define shift-reduce conflict and Reduce-reduce conflict.  
  
A shift-reduce conflict is a conflict that occurs when the parser can either shift the next input symbol onto the stack or reduce the top two symbols on the stack. A reduce-reduce conflict is a conflict that occurs when the parser can reduce the top two symbols on the stack in two different ways.