Homework 1

Keizou Wang

February 6, 2025

1. Errors in a computer program can be classified according to when they are detected and, if they are detected at compile time, what part of the compiler detects them. Using your favorite programming language, give an example of:

(a) A lexical error, detected by the scanner.

```
#include <iostream>
int main(){
    in x[] = {1, 2, 3}; // "in" is an invalid token
    std::cout << x[0] << std::endl;
}
```

(b) A syntax error, detected by the parser.

```
#include <iostream>
int main(){
    int {1, 2, 3} = x[]; // value is left of the assign operator
    std::cout << x[1] << std::endl;
}
```

(c) A static semantic error, detected (at compile-time) by semantic analysis.

```
#include <iostream>
int main(){
    std::cout << x[2] << std::endl; // x is not defined yet
    int x[] = {1, 2, 3};
}
```

(d) A dynamic semantic error, detected (at run-time) by code generated by the compiler.

```
#include <iostream>
int main(){
    int x[] = {1, 2, 3};
    std::cout << x[3] << std::endl; // index 3 is out of bounds
}
```

2. For each of the following languages, write a regular expression that describes the language.

(a) The set of strings that begin with ab and end with ba, over alphabet a, b. Note: the string aba is in the language.

```
a b (a* b)* a
```

(b) The set of natural numbers divisible by 5.

```
digit = 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
5 | digit (digit | 0)* (5 | 0)
```

(c) The set of strings that consist of an even number of a's (including none), over alphabet a.

```
(a a)*
```

3. For each of the following languages, write a grammar that describes the language.

(a) The set of strings that begin with ab and end with ba, over alphabet a, b. Note: the string aba is in the language.

A $\rightarrow$ aba | abBba
B $\rightarrow$ BB | a | b | ϵ

(b) Strings that consist of a sequence of a's followed by a sequence of b's, where the number of a's is even, and equal to the number of b's, over alphabet $\{a, b\}$. In other words, the language $\{a^n b^n \mid n \geq 0 \text{ and } n \text{ is even}\}$.

A $\rightarrow$ aaAbb | ϵ

(c) The set of strings of parentheses (), brackets [], and braces {} that are properly nested. For instance, ()[{}()] is properly nested, while ([)] is not.

A $\rightarrow$ () | [] | {} | (A) | [A] | {A} | AA

4. Consider the following grammar, where P is the start symbol:

P $\rightarrow$ [B, P] | B
B $\rightarrow$ D | (P)
D $\rightarrow$ x | y | z

For each of the following strings, specify whether the string belongs to the language generated by the grammar, and if so, indicate a derivation:

(a) z

The string belongs to the language generated by the grammar.
P $\rightarrow$ B $\rightarrow$ D $\rightarrow$ z

(*b*) (x)

The string belongs to the language generated by the grammar.
P → B → (P) → (B) → (D) → (x)

(*c*) [y]

The string does not belong to the language generated by the grammar.

(*d*) ([x, y])

The string belongs to the language generated by the grammar.
P → B → (P) → ([B, P]) → ([D, P]) → ([x, P]) → ([x, B]) → ([x, D]) → ([x, y])

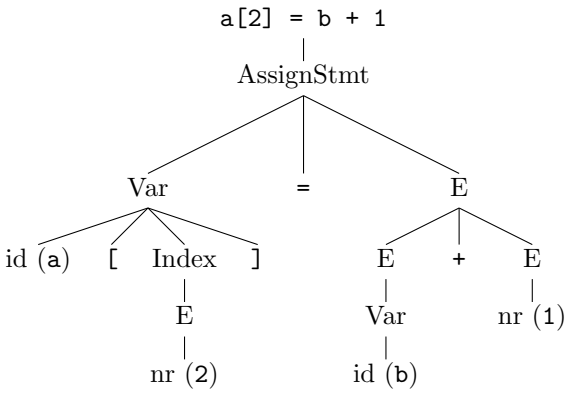
(*e*) [(x), y]

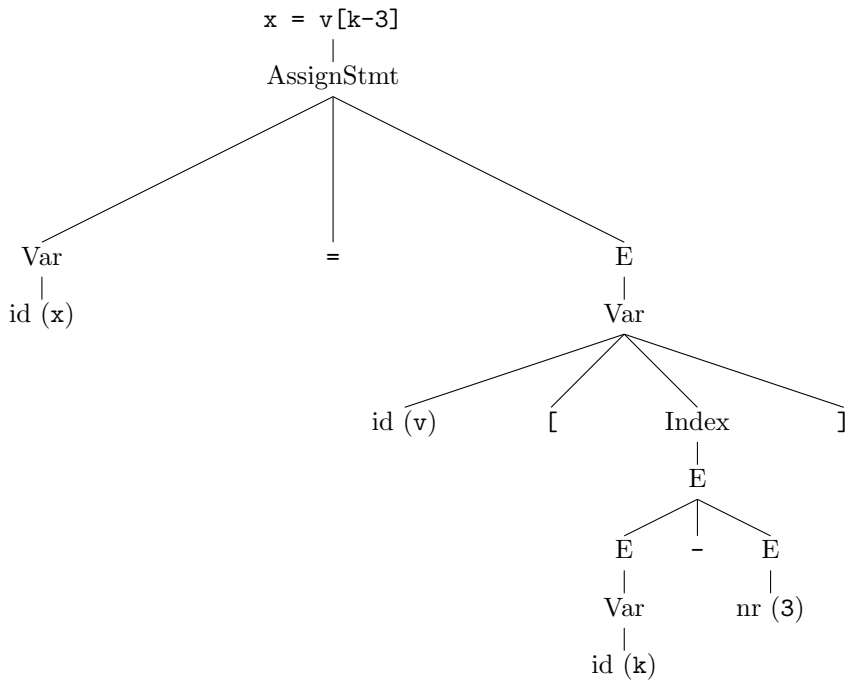
The string belongs to the language generated by the grammar.
P → [B, P] → [(P), P] → [(B), P] → [(D), P] → [(x), P] → [(x), B] → [(x), D] → [(x), y]

5. Consider the following grammar for simple assignment statements:

AssignStmt	→	Var = E
Var	→	id | id [Index]
Index	→	E
E	→	Var | nr | E + E | E − E | E ∗ E | E / E | (E)

(*a*) Build a parse tree for each of the following assignment statements:
`a[2] = b + 1`
`x = v[k-3]`





(b) Rewrite the grammar so that it allows multi-dimensional arrays (instead of just one index, allow a list of indices separated by commas). For instance, each of the following should be a valid string under the new grammar:

```
m[5,3] = x
y = p[i+1,j,k-1]
```

AssignStmt	$\rightarrow$	Var = E
Var	$\rightarrow$	id | id [Index]
Index	$\rightarrow$	E | E, Index
E	$\rightarrow$	Var | nr | E + E | E − E | E ∗ E | E / E | (E)

6. (Extra Credit) Write a regular expression that describes the following language: the set of strings that that contain an even number of a's (including none), not necessarily adjacent, over alphabet $\{a, b\}$.

```
b* (a b* a b*)*
```