CECS 424 – Spring 2018

Assignment 1

Direction: Please complete the following question and submit a PDF file to Dropbox on Beachboard.

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 imperative language C, give an example of each of the following.

(a) A lexical error, detected by the scanner

(b) A syntax error, detected by the parser

(c) A static semantic error, detected by semantic analysis

(d) A dynamic semantic error, detected by code generated by the compiler

(e) An error that the compiler can neither catch nor easily generate code to

catch (this should be a violation of the language definition, not just a

program bug)

2. In your local implementation of C, what is the limit on the size of integers? What happens in the event of arithmetic overflow? What are the implications of size limits on the portability of programs from one machine/compiler to another? How do the answers to these questions differ for Java? For Ada? For Pascal? For Scheme? (You may need to find a manual.)

3. Imperative languages like Fortran and C are typically compiled, while scripting languages, in which many issues cannot be settled until run time, are typically interpreted. Is interpretation simply what one “has to do” when compilation is infeasible, or are there actually some advantages to interpreting a language, even when a compiler is available?

4. Consider the following gcd program written in C

int main() {

int i = getint(), j = getint();

while (i != j) {

if (i > j) i = i % j;

else j = j % i;

}

putint(i);

}

Does this program compute the same result? If not, can you fix it?