CS2510 Practicals

1. Python. Simple syntax, library support, easy to understand.
2. PHP laravel. A massive community with laracasts, stable with activerecord, TDD. Great backen.
3. Measure efficiency- CPU cycles, time. Correctness is almost always more important than efficiency.
4. Cost- Cheap, most developers know JAVA, also it is used pretty much everywhere from GUI to web to database to imbedded systems. Cons- Can be lacking in modern tech implementations due to its age.
5. Unix is programmed in C
6. Yes, except for some specific applications in regard to legacy applications that are cheaper and easier to maintain than rebuild.
7. Millenial bug- programmers coded the year with only two digits- the last two, which made 1900 equivalent to 2000. To save space years omitted the first two digits in punched cards. COBOL PICTURE clause should be used to show dates.
8. JAVA- very first use was for interactive television, but quickly moved on to web based java apps. Popularity: simple, OO, write once and forget, high performance, platform independence, APIs, support
9. ….

# Tasks:

! sum.f90

! Performs summations using in a loop using EXIT statement

program summation

implicit none

integer :: sum, a, count

sum = 0

count = 0

print\*, "Enter 5 for exactly 5 sums, or press any other key."

read\*, a

if (a == 5) then

print\*, "Selected adding exactly 5 sums."

do

print\*, "Add:"

read\*, a

if (a == 0) then

exit

else

if (count == 4) then

exit

else

sum = sum + a

count = count + 1

end if

end if

end do

else

print\*, "This program performs summations. Enter 0 to stop."

do

print\*, "Add:"

read\*, a

if (a == 0) then

exit

else

sum = sum + a

count = count + 1

end if

end do

end if

print\*, "Summation =", sum

print\*, "Count =", count

end

Practical 2

1. Clashes with the English language, making code readable and understandable.
2. Semantics: the meaning of the expressions, statements and programming units. Worry about it so that it makes sense.
3. EBNF is a code that expressed the grammar of a formal language.

**digit excluding zero** = "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" ;

**digit** = "0" | **digit excluding zero** ;

**natural number** = **digit excluding zero**, { **digit** } ;

**digit excluding zero** = "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" ;

**digits even** = {digit\*}, "0" | "2" | "4"| "6" | "8";