

Name: Riley Payung

CDS 251 Midterm
Due: March ~~17~~²³, 2020
By MIDNIGHT uploaded to Blackboard

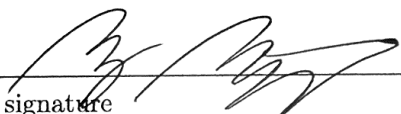
On your Honor!

- Complete this exam in a single sitting.
 - Open notes but **no collaboration, no internet.**
 - Do binary by hand, you may use a calculator for decimal calculations but no 'computer science' calculators (or google) to do binary operations or conversions for you.
 - Exam is designed for a 1 hr 10 min class period. You may take as long as you like as long as you do the exam in a **single sitting**.
 - After you complete the exam, you will not return to it.
 - Do not discuss the exam with anyone until after the due date.
 - Fill in the information below after completing the exam.
-

Date exam taken: 3/23/2020

Time spent on exam: ~~1~~ 55 mins

I attest that I have adhered to the above guidelines:


Your signature

Name: _____

CDS 251 Midterm Spring, 2020

1) (5 pts) a:) What range can a **signed** byte have and b:) how many total integers can one byte represent?

a) signed Byte: -128 to 127

b) ~~255~~ 256 (-128 - 127) or (0 to 255)

2) (10 pts) **Signed** binary addition. Add in binary. State the error if there is one. If no error, convert answer to decimal.

$$\begin{array}{r} \overset{1}{1} \overset{1}{1} \overset{1}{1} \overset{1}{1} \overset{1}{1} \overset{1}{0} \overset{1}{1} \overset{1}{1} \\ + 0 0 1 0 1 1 0 1 \\ \hline 1 0 0 1 0 1 0 0 0 \end{array}$$

Overflow error (9th bit)

3) (10 pts) **Unsigned** binary addition. Add in binary. State the error if there is one. If no error, convert to decimal.

$$\begin{array}{r} \overset{1}{0} \overset{1}{1} \overset{1}{1} \overset{1}{0} \overset{1}{1} \overset{1}{1} \overset{1}{0} \overset{1}{1} \\ + 0 0 1 1 1 1 0 0 \\ \hline 1 0 1 0 1 0 0 1 \end{array}$$

$$\begin{array}{r} 41 \\ + 128 \\ \hline 169 \end{array}$$

169. No error.

4) (10 pts) The following is a four byte Single Precision Real Number. What is the number in decimal?

1 10000111 101000000000000000000000

$$\begin{aligned}
 & 1 + 0.25 = 1.25 \\
 & 1.25 \times 2^{-7} = 0.009765625 \\
 & 1 + 0.25 = 1.25 \times 2^{-7} = 0.009765625
 \end{aligned}$$

5) (5 pts) I have a real number set as : Number = 2.34×10^{-43} . I divide this number by 300. The result in Fortran is zero. Why did this happen and what is it called?

Since we were dividing a real by an integer, we lose the precision of the real, and default to an integer.

This is called Lack of Precision or Precision Error.

6) (5 pts) You store the x, y, and z coordinates of stars in a galaxy simulator in an array declared like this: (double precision uses 8 bytes of memory)

double precision :: Stars(20000, 3)

How many bytes apart in computer memory are the x and y coordinates of any particular star stored?

$$20000 \times 8 = 160000 \text{ bytes apart.}$$

7) (5 pts) Re-write the declaration for the Stars array so that there would be no cache misses when accessing the x, y, and z coordinates of a star. (Just rewrite the one declaration line for the array.)

double precision :: Stars(3, 20000)

8) (5 pts) Write the **update formula** for computing Average as you are reading in data. Write just the one line as you would have it in a program.

$$a = a + \frac{x(i) - a}{i}$$

$$a = a + ((x(i) - a) / \text{float}(i))$$

9) (5 pts) $\sigma = \sqrt{E[X^2] - (E[X])^2}$ Give two reasons why this is a particularly bad way to compute standard deviation for a large data set.

- The square root function is very slow
- Higher risk of overflow
- Loss of precision

10) (5 pts) Recursive Fibonacci and Recursive Quick Sort are both Multiple Recursive algorithms. Which one is good? Which one is bad? Explain.

Recursive Fibonacci is bad because it recursively re-runs numbers that have already been run, creating large hang time. Quicksort is good because ~~it~~ in each iteration, half of the work remains.

11) (10 pts) Say QuickSort takes 1.1 seconds to sort 60,000 numbers and Bubble Sort takes 36.9 seconds. Calculate how long would each one take to do 18 million numbers? **Answer in appropriate units!**

$$\begin{aligned} 36.9 \times (60000)^2 &= 132840 \text{ seconds} \\ 36.9 \times 18000000 &= 664200000 \text{ seconds} \\ 36.9 \times 300^2 &= 3321000 \text{ seconds} \\ &= 38.44 \text{ days} \end{aligned}$$

$$\begin{aligned} &\frac{60000}{18000000} \swarrow \searrow \\ &\frac{18000000}{60000} = 300 \end{aligned}$$

12) (5 pts) Write Fortran code that will swap the values in variables B and C . (Just the swapping code, no declarations or anything else.)

```
temp = B
B = C
C = temp
```

13) (5 pts) Write a do loop that prints the odd numbers from 5 to 51 to the screen. Just write the do loop, no declarations or any other stuff.

```
do i = 5, 51, 2  
  print*, i
```

```
enddo
```

14) (15 pts) Write a **recursive function** that returns the sum of the numbers 1 to n. All variables are to be declared and **good programming** techniques are to be used inside the function. The function is called from some program (**do not write this program**) in a line like this:

MySum = SumOfNumbers(n)

~~recursive function funsum(n) result(m)~~

recursive function SumOfNumbers(n) result (sumA)

~~integer~~ integer :: n, sumA

if (n == 1) then

~~return~~ sumA = sumA + 1
return

else

sumA = sumA + SumOfNumbers(n-1)

endif

end function ~~fun~~ SumOfNumbers