537 Exam 1

Fall 2025

Tuesday, September 30, 2025

Name: Solutions

Person number:

0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9

537 Fall 2025

EXAM INSTRUCTIONS

The exam is closed-book. No calculator.
One single side of handwritten notes is permitted.

- **Relax.**
- **ANSWER ANY 4 OF THE 5 QUESTIONS**
- **Work steadily and carefully.**
- **You have 20 minutes per question.**
  If you get bogged down on a question, move on.
- **For maximum credit, show all your work.**
- **Do not waste time writing information that is not asked for.**
- **About 5% will be awarded for "style":**
  be sure your writing is easy to read, and
  make your reasoning, explanations, and calculations
  clear, explicit, easy to follow.

1

(a) i. How many distinct floating-point numbers are there in the IEEE 754 64-bit system?
ii. What fraction of the numbers do we lose by reserving the top exponent for non-numbers (nan, inf)?
Briefly explain your answers.
iii. Also put the following numbers in increasing order, using the fact that $2^p \approx 10^{0.30\,p}$:
S = the number of stars in a typical galaxy ~ 10^{11}
M = the number of molecules in a drop of water ~ 5×10^{20}
N = the number of distinct floating-point numbers in the IEEE 754 64-bit system.

(b) What is the **definition** and the **significance** of the number ε_{mach} for a floating-point arithmetic system.

ε_{mach} is the spacing between 1 and the next machine number. It measures the resolution of the system of machine numbers.
It is the maximum relative spacing between (non-zero) machine numbers,
and twice the maximum relative error in rounding a real number (in range) to a machine number.

IMPORTANT: there is a part (c) on the next page.

1

(c) Small integers are represented exactly in IEEE 754 64-bit floating-point format.

(Recall that this format has a sign bit, an 11-bit exponent biased by $1023 = 2^{10} - 1$, and a 52-bit mantissa.)

Discover what is smallest positive integer that *cannot* be represented exactly**.**

Explain your reasoning clearly,

and also give the decimal order of magnitude of the answer.

For economy suppose for a moment we have a 4-bit mantissa (instead of 52-bit).

Then the small positive integers are represented as follows:

1	1.0000×2^0	=	1
2	1.0000×2^1	=	10
3	1.1000×2^1	=	11
⋮			
$2^{4+1}-1$	1.1111×2^4	=	11111
2^{4+1}	1.0000×2^5	=	100000

All these integers so far are machine numbers.

However,

$2^{4+1}+1 = 100001$ (4 bits)

would require an additional bit in the mantissa to represent exactly.

Thus for a 52-bit mantissa, the answer is

$$2^{52+1}+1 = \boxed{2^{53}+1} \approx 10^{(.3)53} = 10^{15.9} \approx \boxed{10^{16}}.$$

Check:

```
$ ./hexcodeout
Enter number: 9007199254740991
433fffffffffffff
Enter number: 9007199254740992
4340000000000000
Enter number: 9007199254740993
4340000000000000
Enter number: 9007199254740994
4340000000000001
```

```
2**53
9007199254740992

i = float(2**53)

i-1
9007199254740991.0

i
9007199254740992.0

i+1
9007199254740992.0

i+2
9007199254740994.0
```

same

2 The pink text is just for motivation: you can completely ignore it for now. In a round of "musical chairs", let p denote the number of players, and c the number of available chairs. If the players choose seats at random, the expected number of players surviving to the next round is

$$\left(1-e^{-\frac{p}{c}}\right)c$$

(a) Explain why direct evaluation of this formula in finite-precision arithmetic is problematic for $0 < p << c$.
This can be relevant if c, the number of "chairs", is very large, as might be the case if the "players" are insect parasites, and "chairs" are the hosts they parasitize.

For $0<p\ll c$, $e^{-p/c}\approx 1$, so we are subtracting near-equals which we know results in severe loss of significance.

(b) How would you evaluate this function accurately when $p << c$? Be specific. If you introduce any truncation error, it should be $O(p^3)$ or smaller.

Use Taylor approx for $e^{-p/c}$:

$$e^x = 1 + x + \frac{x^2}{2!} + O(x^3)$$

$$e^{-p/c} = 1 - \frac{p}{c} + \frac{p^2}{2c^2} + O\left(\left(\frac{p}{c}\right)^3\right)$$

and

$$(1-e^{-p/c})c = c\left(1 - \left(1 - \frac{p}{c} + \frac{p^2}{2c^2} + O\left(\left(\frac{p}{c}\right)^3\right)\right)\right)$$

$$= p - \frac{p^2}{2c} + O\left(\left(\frac{p}{c}\right)^3\right)$$

(Contextual sanity check: when there are many more "chairs" than players, almost everyone survives: #survivors $\approx p$.)

3 Find the approximate maximum relative error in the floating point evaluation of this expression

$$(x+c)-c$$

when x is small and *not* necessarily a machine number,
and c is a *machine number* greater than 1.
Answer in terms of $|x|$, c, and ε_{mach}, and just keep the term or terms that dominate as $|x|$ goes to zero.
Suggestion: to avoid mistakes, expand everything fully before canceling anything.

$$\left|\text{relative error}\right| = \left| \frac{\left[\left(\overbrace{x(1+\delta_1)+c}^{fl(x)}\right)(1+\delta_2) - c\right](1+\delta_3) - x}{x} \right|$$

$$= \left| \frac{\left(x + x\delta_1 + c + x\delta_2 + x\delta_1\delta_2 + c\delta_2 - c\right)(1+\delta_3) - x}{x} \right|$$

$$= \left| \frac{x + x\delta_1 + x\delta_2 + x\delta_1\delta_2 + c\delta_2 + x\delta_3 + x\delta_1\delta_3 + x\delta_2\delta_3 + x\delta_1\delta_2\delta_3 + c\delta_2\delta_3 - x}{x} \right|$$

$$= \left| \frac{x}{x}\left(\delta_1 + \delta_2 + \delta_1\delta_2 + \delta_3 + \delta_1\delta_3 + \delta_2\delta_3 + \delta_1\delta_2\delta_3\right) + \frac{c\delta_2 + c\delta_2\delta_3}{x} \right|$$

$$\approx \left| (\delta_1 + \delta_2 + \delta_3) + \frac{c\delta_2}{x} \right|$$

$$\leq 3\varepsilon_{mach} + \frac{c\varepsilon_{mach}}{2|x|} \xrightarrow{x\to 0} \frac{c\varepsilon_{mach}}{2|x|}.$$

(catastrophic error as $x \to 0$).

4

Dividing by multiplying

Suppose you have a computer chip that can do multiplication but not division.Consider working around the deficit by obtaining the reciprocal of a real number $r \neq 0$ by finding a root of an appropriate function using Newton's method.

Each of the following two functions has a root at $1/r$:

(i) $f_1(x) = rx - 1$

(ii) $f_2(x) = 1 - \frac{1}{rx}$

(a) Which would be satisfactory for the purpose described above? Explain in detail.

(b) For the one that works, what is the order of convergence? Justify your answer by citing an appropriate theorem.

(a) Newton iteration is $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$.

(i) $f_1'(x) = r$

So $x_{k+1} = x_k - \frac{(rx_k - 1)}{r} = \frac{1}{r}$.

So Newton converges to $\frac{1}{r}$ immediately.
But division is required to compute $\frac{1}{r}$.
So not useful.

(ii) $f_2'(x) = +\frac{1}{rx^2}$,

So $x_{k+1} = x_k - \frac{\left(1 - \frac{1}{rx_k}\right)}{\left(\frac{1}{rx^2}\right)} = x_k - rx_k^2 + x_k$

$= 2x_k - rx_k^2$.

This can be computed without division!

(b) $f_2'\left(\frac{1}{r}\right) = r \neq 0$ and $f_2' \in C^1(\mathbb{R}^+)$.

So by Thm 2.4 or Thm 2.5 Newton on f_2 converges (at least) quadratically.

Test Newton on f_2:

```
def reciprocal(r):
    # crude starting guesses
    if r>1:
        x = .5
    else:
        x = 2
    tol = 1e-15 # relative error tolerance
    i = 0
    print(i,x)
    while True:
        s = x - r*x*x
        x += s
        i += 1
        print(i,x)
        if abs(r*s) <= tol: return x
        if i>8:
            print('convergence not achieved')
            break
```

```
reciprocal(3)
```

```
0 0.5
1 0.25
2 0.3125
3 0.33203125
4 0.3333282470703125
5 0.3333333332557231
6 0.3333333333333333
7 0.3333333333333333

0.3333333333333333
```

```
reciprocal(.1)
```

```
0 2
1 3.6
2 5.904
3 8.3222784
4 9.718525023289343
5 9.992077183748574
6 9.999993722898264
7 9.99999999999606
8 9.999999999999998
9 10.0

10.0
```

5

Convergence of functional iteration

Suppose a function $g : \mathbb{R} \to \mathbb{R}$ has the following properties:

- $g(z) = z$,
- g is continuously differentiable on an open interval containing z, and
- $|g'(z)| < 1$.

Use the Mean Value Theorem to prove that there is an interval I about z such that g is a *contraction* on I **and** g maps I into itself (so that the Contraction Mapping Theorem guarantees convergence to z for a sufficiently close starting point).

Since $|g'(z)| < 1$ and g' is continuous, we can choose an $\varepsilon > 0$ (small enough) such that on the interval $I = [z-\varepsilon, z+\varepsilon]$, $|g'| \leq L < 1$.

Then for x, y in I, the Mean Value Theorem says

$$|g(x) - g(y)| < |g'(\xi)(x-y)| \leq L|x-y|.$$

That is, g is L-Lipschitz with $L < 1$, which is the definition of a contraction.

Then for $x \in I$,

$$|g(x) - z| = |g(x) - g(z)| \leq L|x-z| \leq L\varepsilon < \varepsilon$$

Thus $g(x) \in I$, that is, g maps I into itself.