

1. **Simplify and find a function g such that $f(x) = O(g(x))$.**

Given:

$$f(x) = (x^{2.3} + x \ln(x^5)) (1.1x + 1 + 1.2x) + (x^2 + 1.2x) (x^3 + 0.92x).$$

Step 1:

$$x^{2.3} + x \ln(x^5) = O(x^{2.3}), \quad 1.1x + 1 + 1.2x = O(x).$$

Their product is $O(x^{3.3})$.

Step 2:

$$x^2 + 1.2x = O(x^2), \quad x^3 + 0.92x = O(x^3).$$

Their product is $O(x^5)$.

Step 3: Add the two results:

$$f(x) = O(x^{3.3}) + O(x^5).$$

Step 4: Since x^5 dominates $x^{3.3}$ for large x ,

$$f(x) = O(x^5).$$

$$\textbf{Answer:} g(x) = x^5.$$

2. **Is there a smallest real number a such that $x^2 6^x$ is $O(a^x)$?**

(a) If $x^2 6^x \leq C a^x$ for large x , then $(\frac{6}{a})^x$ must eventually shrink faster than x^{-2} .

(b) This requires $\frac{6}{a} < 1$, or $a > 6$.

(c) No real number $a \leq 6$ can work, and there is no minimum real > 6 .

Answer:

No. We must have $a > 6$, but there is no smallest real number.

3. **Hex addition CA10 + 4F57.**

CA10

+ 4F57

(a) Add the rightmost digits:

$$0 + 7 = 7.$$

(b) Next digits:

$$1 + 5 = 6.$$

(c) Next, add:

$$A_{16} + F_{16} = 10_{10} + 15_{10} = 25_{10}.$$

In hexadecimal, $25_{10} = 19_{16}$; write down 9 and carry 1.

(d) Finally, add the leftmost digits with the carry:

$$C_{16} + 4_{16} + 1(\text{carry}) = 12_{10} + 4_{10} + 1_{10} = 17_{10}.$$

Since $17_{10} = 11_{16}$, write down 1 and carry 1 (which becomes a new digit at the front).

Answer: 11967_{16}

In hexadecimal addition, when the sum of digits (and any carry) exceeds F_{16} , convert the total sum to hexadecimal. For example, $1+1+1$ in binary yields 1 with a carry of 1 because the result exceeds the single-digit value in that base.

4. **Binary multiplication** 11011×1001 .

$$\begin{array}{r} 11011 \\ \times 1001 \\ \hline 11011 \\ 000000 \\ 0000000 \\ +11011000 \\ \hline 11110011 \end{array}$$

In binary addition, when adding three 1's together, $1 + 1 + 1$, the result is 1 with a carry of 1 since binary digits are only 0 and 1.

Answer: 11110011_2

5. **Use the Euclidean Algorithm to show** $\gcd(621, 82) = 1$.

Steps:

(a) $621 = 82 \cdot 7 + 47$; $\gcd(621, 82) = \gcd(82, 47)$.

(b) $82 = 47 \cdot 1 + 35$; $\gcd(82, 47) = \gcd(47, 35)$.

(c) $47 = 35 \cdot 1 + 12$; $\gcd(47, 35) = \gcd(35, 12)$.

(d) $35 = 12 \cdot 2 + 11$; $\gcd(35, 12) = \gcd(12, 11)$.

(e) $12 = 11 \cdot 1 + 1$; $\gcd(12, 11) = \gcd(11, 1)$.

(f) $11 = 1 \cdot 11 + 0$; $\gcd(11, 1) = 1$.

Answer: $\gcd(621, 82) = 1$.

6. **Show that you can multiply n by 35 using only five multiplications by 2 and two additions**

Steps:

(a) $35_{10} = 32 + 2 + 1 = 2^5 + 2^1 + 2^0$.

(b) Double n five times to get $32n$.

(c) Then add $(32n) + (2n) + n = 35n$.

Answer: $35n = (2^5 n) + (2n) + n$.

7. **Multiply an octal number n by $(10)_8$. Illustrate with $(741)_8$.**

Multiplying any number by its base shifts the digits one place to the left.
In octal:

$$(10)_8 = 8 \quad (\text{in decimal}).$$

Thus, multiplying $(741)_8$ by $(10)_8$ shifts every digit one place to the left:

$$(741)_8 \times (10)_8 = (7410)_8.$$

Answer: 7410_8 .

8. **Divide an octal number n by $(10)_8$. Illustrate with $(741)_8$.**

Steps:

(a) In octal, dividing by $(10)_8$ is the same as removing the last digit as remainder.

(b) Example: $(741)_8 \div (10)_8$ equals $(74)_8$ and remainder $(1)_8$.

Answer: $(74)_8$, remainder $(1)_8$.

9. **Fast exponentiation for $32^n \bmod 13$.**

Step 1: Reduce the Base

$$32 \div 13 = 2 \quad \text{with remainder } 32 - 2 \cdot 13 = 32 - 26 = 6,$$

$$32 \equiv 6 \pmod{13}.$$

Thus, for any exponent n ,

$$32^n \equiv 6^n \pmod{13}.$$

Step 2: Verify for Small Exponents

- For $n = 0$:

$$6^0 = 1.$$

- For $n = 1$:

$$6^1 = 6.$$

- For $n = 2$:

$$6^2 = 36.$$

Dividing 36 by 13:

$$36 \div 13 = 2 \quad (\text{since } 13 \times 2 = 26), \quad 36 - 26 = 10,$$

so

$$6^2 \equiv 10 \pmod{13}.$$

Step 3: Apply Euler's Theorem

Because 13 is prime, its Euler totient is

$$\varphi(13) = 13 - 1 = 12.$$

Since 6 and 13 share no common factors other than 1 (coprime), Euler's theorem tells us:

$$6^{12} \equiv 1 \pmod{13}.$$

This means the powers of 6 modulo 13 are periodic with period 12.

Step 4: Reduce the Exponent

For any exponent n , we can reduce it modulo 12:

$$6^n \equiv 6^{n \bmod 12} \pmod{13}.$$

$$\textbf{Answer: } 6^{n \bmod 12} \pmod{13}.$$

10. Closed-form for the hex number of n copies of 'A'.

Step 1: Express the Number as a Sum

An n -digit number where each digit is A can be written as

$$\underbrace{AA \cdots A}_{n \text{ digits}} = \sum_{k=0}^{n-1} A \cdot 16^k.$$

Since $A = 10$, this becomes

$$\sum_{k=0}^{n-1} 10 \cdot 16^k = 10 \sum_{k=0}^{n-1} 16^k.$$

Step 2: Sum the Geometric Series

The sum of the geometric series with n terms and common ratio 16 is given by:

$$\sum_{k=0}^{n-1} 16^k = \frac{16^n - 1}{16 - 1} = \frac{16^n - 1}{15}.$$

This becomes:

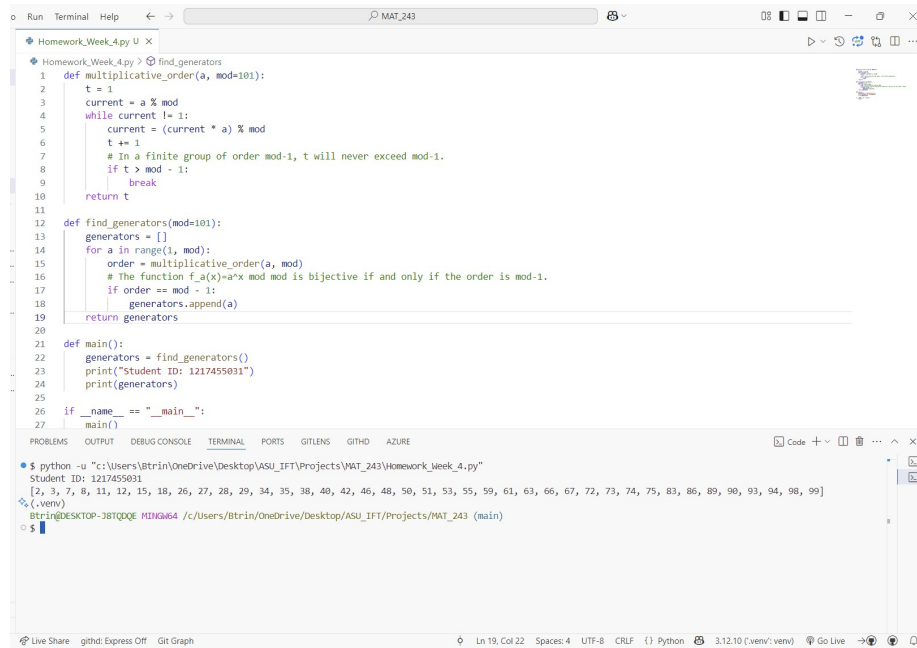
$$10 \cdot \frac{16^n - 1}{15}.$$

Step 3: Simplify the Expression

$$10 \cdot \frac{16^n - 1}{15} = \frac{10(16^n - 1)}{15} = \frac{2(16^n - 1)}{3}.$$

$$\textbf{Answer: } \frac{2(16^n - 1)}{3}.$$

11. Extra Credit



The screenshot shows a VS Code editor window with a file named `Homework_Week_4.py`. The code defines two functions: `multiplicative_order` and `find_generators`, and a `main` function. The `multiplicative_order` function finds the smallest positive integer `t` such that $a^t \equiv 1 \pmod{mod}$. The `find_generators` function finds all integers `a` in the range $1 \leq a < mod$ such that a is a generator modulo `mod`. The `main` function calls `find_generators` and prints the student ID and the list of generators.

```
1 def multiplicative_order(a, mod=101):
2     t = 1
3     current = a % mod
4     while current != 1:
5         current = (current * a) % mod
6         t += 1
7         # In a finite group of order mod-1, t will never exceed mod-1.
8         if t > mod - 1:
9             break
10    return t
11
12 def find_generators(mod=101):
13     generators = []
14     for a in range(1, mod):
15         order = multiplicative_order(a, mod)
16         # The function f_a(x)=a*x mod mod is bijective if and only if the order is mod-1.
17         if order == mod - 1:
18             generators.append(a)
19    return generators
20
21 def main():
22     generators = find_generators()
23     print("Student ID: 1217455031")
24     print(generators)
25
26 if __name__ == "__main__":
27     main()
```

The terminal output shows the execution of the script:

```
$ python -u "c:\Users\Btr\OneDrive\Desktop\ASU_IFT\Projects\WAT_243\Homework_Week_4.py"
Student ID: 1217455031
[2, 3, 7, 8, 11, 12, 15, 18, 26, 27, 28, 29, 34, 35, 38, 40, 42, 46, 48, 50, 51, 53, 55, 59, 61, 63, 66, 67, 72, 73, 74, 75, 83, 86, 89, 90, 93, 94, 98, 99]
```