TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH

BÀI TẬP MÔN PHÂN TÍCH VÀ THIẾT KẾ THUẬT TOÁN
HOMEWORK #03: ĐỘ PHỨC TẠP VÀ CÁC KÍ HIỆU TIỆM CẬN

GV hướng dẫn: Huỳnh Thị Thanh Thương

Nhóm thực hiện:
1. Nguyễn Gia Bảo - 22520109
2. Phạm Nguyên Anh - 22520069
3. Phạm Huỳnh Nhựt Anh - 22520068
4. Hoàng Công Chiến - 22520155

TP.HCM, ngày 13 tháng 04 năm 2024

Câu 1:

Bảng 1: Thời gian chạy của các hàm

Hàm	**Đơn vị thời gian**			
	1 giây	**1 phút**	**1 giờ**	**1 ngày**
$\log n$	2^{10^6}	$2^{60.10^6}$	$2^{36.10^8}$	$2^{864.10^8}$
$\sqrt{n}$	10^{12}	$3,6.10^{15}$	$1,296.10^{19}$	$7,46496.10^{21}$
n	10^6	60.10^6	36.10^8	864.10^8
$nlogn$	62746	2801416	133378057	2755147512
n^2	1000	7745	60000	293938
n^3	100	391	1532	4420
2^n	19	25	31	36
$n!$	9	11	12	13

Bảng 2: Thời gian chạy của các hàm

Hàm	**Đơn vị thời gian**		
	1 tháng	**1 năm**	**1 thế kỷ**
$\log n$	$2^{2592.10^9}$	$2^{31536.10^9}$	$2^{31536.10^{11}}$
$\sqrt{n}$	$6,718464.10^{24}$	$9,94519296.10^{26}$	$9,94519296.10^{30}$
n	2592.10^9	31536.10^9	31536.10^{11}
$nlogn$	71870856404	797633893349	68610956750569
n^2	1609968	5615692	56156922
n^3	13736	31593	146645
2^n	41	44	51
$n!$	15	16	17

Code giúp tính toán giá trị của n:

cs112.cpp

cs112.cpp > calculate(double, string)

```cpp
#include<bits./stdc++.h>
#include<iomanip>

using namespace std;

long long giaithua(long long n)
{
    if (n==1) return 1;
    return (n)*giaithua(n-1);
};

double calculate(double time, string func)
{
    double i = 0;
    double res = 0;
    if (func == "logn")
    {
        return floor(pow(2,time));
    }

    if (func == "sqrt(n)")
    {
        return time*time;
    }

    if (func == "n")
    {
        return time;
    }

    if (func == "nlogn")
    {
        double left = 0;
        double right = time*pow(10,6);
        double  mid = 0;
        while (true)
        {
            if (left <= right)
            {
                mid  = ceil((left + right)/2);
```

cs112.cpp

cs112.cpp > calculate(double, string)

```
12  double calculate(double time, string func)
39              {
40                  mid  = ceil((left + right)/2);
41                  res = (mid * log2(mid));
42
43                  // cout << left << ":" << mid << ":" << right << " --> " << res
44                  if (res > time)
45                  {
46                      right = mid -1;
47                  }
48                  else if (res < time)
49                  {
50                      left = mid + 1;
51                  }
52              }
53              else
54              {
55                  break;
56              }
57          }
58
59          return mid-1;
60      }
61
62      if (func == "n^2")
63      {
64          return floor(sqrt(time));
65      }
66
67      if (func == "n^3")
68      {
69          i = floor(pow(time, 1.0/3.0));
70          if (pow(i+1,3) <= time)
71          {
72              return i + 1;
73          }
74          else
75          return i;
76      }
77
```

cs112.cpp

cs112.cpp > calculate(double, string)

```cpp
double calculate(double time, string func)
    if (func == "2^n")
    {
        return floor(log2(time));
    }
    if (func == "n!")
    {
        res = 1;
        i = 1;
        while (res <= time)
        {
            i++;
            res = giaithua(i);
        }
    }

    return i-1;

};

void print_(vector<double> time, string func){
    cout << "With " << func << ":" << endl;
    cout << fixed << setprecision(2);
    if (func == "logn")
    {
        cout << "    In 1 second, max n = 2^"  << (time[0]) <<  endl;
        cout << "    In 1 minute, max n = 2^"  << (time[1]) <<  endl;
        cout << "    In 1 hour, max n = 2^"  << (time[2]) <<  endl;
        cout << "    In 1 day, max n = 2^"  << (time[3]) <<  endl;
        cout << "    In 1 day, max n = 2^"  << (time[4]) <<  endl;
        cout << "    In 1 year, max n = 2^" <<  (time[5]) << endl;
        cout << "    In 1 century, max n = 2^" <<  (time[6]) << endl;
        return;
    }

    double second_n = calculate(time[0], func);
    cout << "    In 1 second, max n = "  << second_n <<  endl;

    double minute_n = calculate(time[1], func);
    cout << "    In 1 minute, max n = "  << minute_n <<  endl;
```

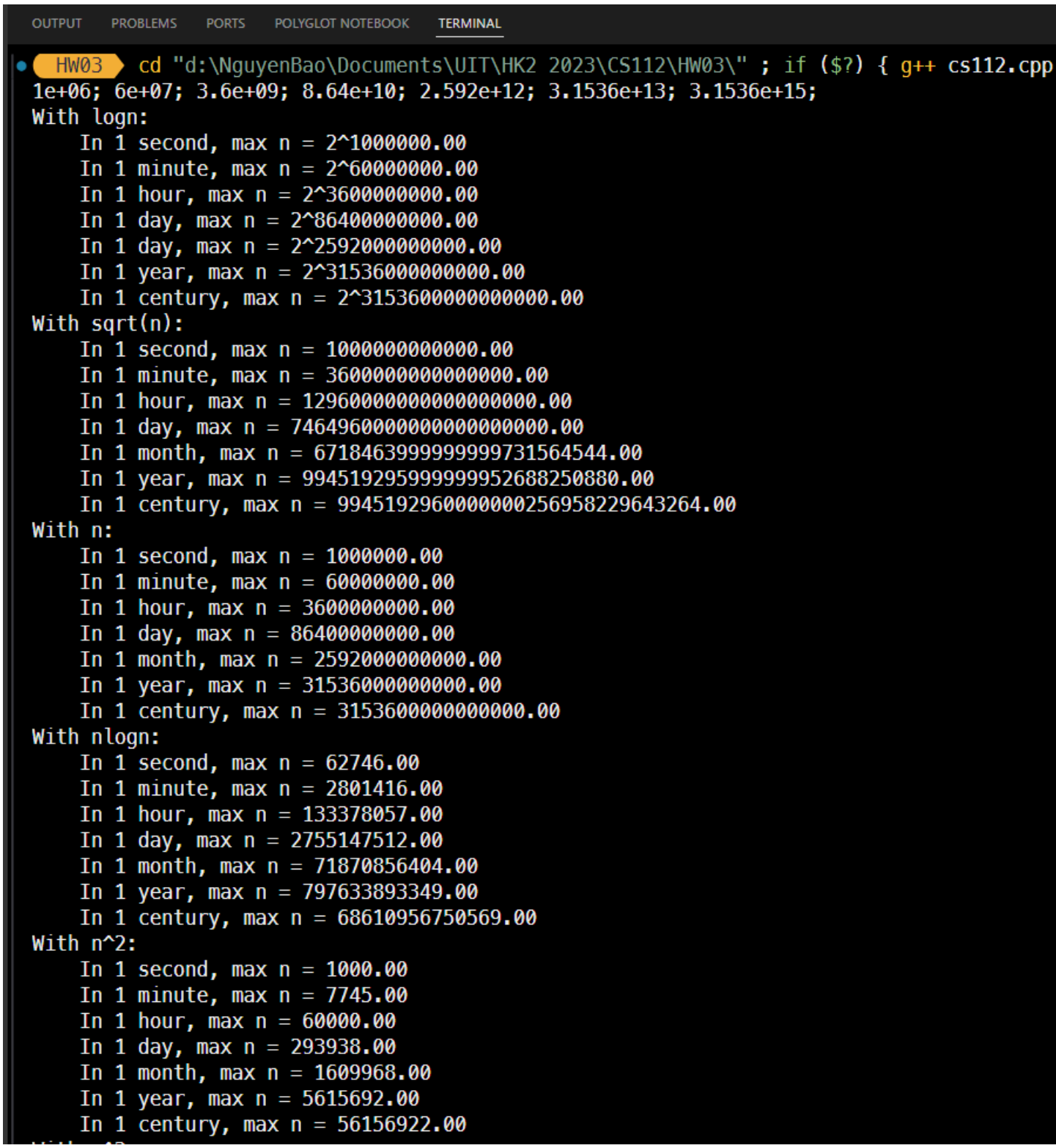
cs112.cpp

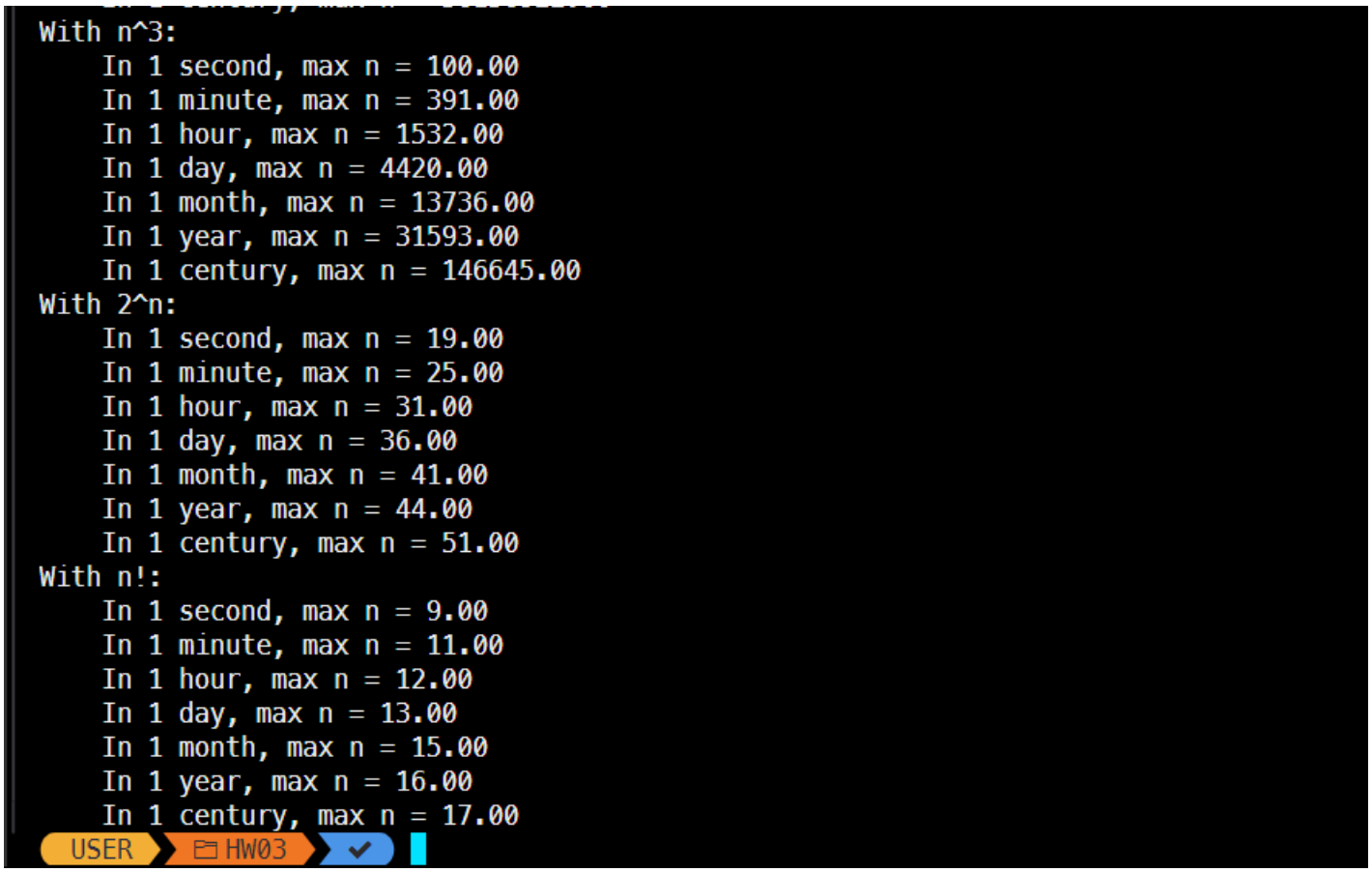
```cpp
void print_(vector<double> time, string func){
    double minute_n = calculate(time[1], func);
    cout << "    In 1 minute, max n = "  << minute_n <<  endl;

    double hour_n = calculate(time[2], func);
    cout << "    In 1 hour, max n = "  << hour_n <<  endl;

    double day = calculate(time[3], func);
    cout << "    In 1 day, max n = "  << day <<  endl;

    double month = calculate(time[4], func);
    cout << "    In 1 month, max n = "  << month <<  endl;

    double year_n = calculate(time[5], func);
    cout << "    In 1 year, max n = " <<  year_n << endl;

    double century_n = calculate(time[6], func);
    cout << "    In 1 century, max n = " <<  century_n << endl;

};

int main()
{
    double second = pow(10, 6)  ; // 1second = 10^6 microseconds
    double minute = second * 60;
    double hour = minute * 60;
    double day = hour * 24;
    double month = day * 30;
    double year = day * 365; // 1 year = 365 days
    double century = year * 100; // 1 century = 100 years

    vector<double> time = {second, minute, hour, day, month, year, century};

    for (auto t : time){
        cout << t << "; ";
    }
    cout << endl;
    print_(time, "logn");
    print_(time, "sqrt(n)");
    print_(time, "n");
```

cs112.cpp

```
cs112.cpp > print_(vector<double>, string)
 97  void print_(vector<double> time, string func){
125      cout << "    In 1 month, max n = "  << month <<  endl;
126
127      double year_n = calculate(time[5], func);
128      cout << "    In 1 year, max n = " <<  year_n << endl;
129
130      double century_n = calculate(time[6], func);
131      cout << "    In 1 century, max n = " <<  century_n << endl;
132
133  };
134
135  int main()
136  {
137      double second = pow(10, 6)  ; // 1second = 10^6 microseconds
138      double minute = second * 60;
139      double hour = minute * 60;
140      double day = hour * 24;
141      double month = day * 30;
142      double year = day * 365; // 1 year = 365 days
143      double century = year * 100; // 1 century = 100 years
144
145      vector<double> time = {second, minute, hour, day, month, year, century};
146
147      for (auto t : time){
148          cout << t << "; ";
149      }
150      cout << endl;
151      print_(time, "logn");
152      print_(time, "sqrt(n)");
153      print_(time, "n");
154      print_(time, "nlogn");
155      print_(time, "n^2");
156      print_(time, "n^3");
157      print_(time, "2^n");
158      print_(time, "n!");
159
160
161      return 0;
162  }
```

Dưới đây là kết quả khi chạy chương trình trên:


OUTPUT
PROBLEMS
PORTS
POLYGLOT NOTEBOOK
TERMINAL
HW03
cd "d:\NguyenBao\Documents\UIT\HK2 2023\CS112\HW03\" ; if ($?) { g++ cs112.cpp
1e+06; 6e+07; 3.6e+09; 8.64e+10; 2.592e+12; 3.1536e+13; 3.1536e+15;
With logn:
In 1 second, max n = 2^1000000.00
In 1 minute, max n = 2^60000000.00
In 1 hour, max n = 2^3600000000.00
In 1 day, max n = 2^86400000000.00
In 1 day, max n = 2^2592000000000.00
In 1 year, max n = 2^31536000000000.00
In 1 century, max n = 2^3153600000000000.00
With sqrt(n):
In 1 second, max n = 1000000000000.00
In 1 minute, max n = 3600000000000000.00
In 1 hour, max n = 12960000000000000000.00
In 1 day, max n = 746496000000000000000.00
In 1 month, max n = 671846399999999731564544.00
In 1 year, max n = 994519295999999952688250880.00
In 1 century, max n = 994519296000000025695822964326 4.00
With n:
In 1 second, max n = 1000000.00
In 1 minute, max n = 60000000.00
In 1 hour, max n = 3600000000.00
In 1 day, max n = 86400000000.00
In 1 month, max n = 2592000000000.00
In 1 year, max n = 31536000000000.00
In 1 century, max n = 3153600000000000.00
With nlogn:
In 1 second, max n = 62746.00
In 1 minute, max n = 2801416.00
In 1 hour, max n = 133378057.00
In 1 day, max n = 2755147512.00
In 1 month, max n = 71870856404.00
In 1 year, max n = 797633893349.00
In 1 century, max n = 68610956750569.00
With n^2:
In 1 second, max n = 1000.00
In 1 minute, max n = 7745.00
In 1 hour, max n = 60000.00
In 1 day, max n = 293938.00
In 1 month, max n = 1609968.00
In 1 year, max n = 5615692.00
In 1 century, max n = 56156922.00



Giải thích. với biến time là khoảng thời gian (milisecond) được truyền vào hàm:
- Đối với hàm $\log n$, vì có cơ số 2 nên chúng ta chỉ cần trả về giá trị 2^{time}
- Đối với hàm $\sqrt{n}$, ta chỉ cần trả về giá trị time bình phương.
- Đối với hàm $f(n) = n$, ta chỉ đơn giản trả về giá trị time.
- Đối với hàm $n.\log n$, ta dùng kĩ thuật binary search để tìm kiếm n lớn nhất.
- Đối với hàm n^2 và n^3, ta chỉ cần trả về giá trị căn bậc 2 và bậc 3 của biến time (làm tròn xuống)
- Đối với hàm 2^n, ta trả về giá trị $\log_2 time$
- Đối với hàm $n!$, do hàm này tăng rất nhanh, nên ta có thể dùng cách vét cạn để tìm giá trị n tối đa.

Câu 2: Sắp xếp tăng dần theo Big O nhỏ nhất,giải thích cách so sánh

Group 1

$$f_1(n) = \binom{n}{100}$$
$$f_2(n) = n^{100}$$
$$f_3(n) = \frac{1}{n}$$
$$f_4(n) = 10^{1000}n$$
$$f_5(n) = n \log n$$

Ta có :
$f_1(n) = \binom{n}{100} = \frac{n!}{100!(n-100)!} = \frac{n(n-1)(n-2)...(n-99)}{100!} = O(n^{100})$
$f_2(n) = n^{100} = O(n^{100})$
$f_3(n) = \frac{1}{n}$
$f_4(n) = 10^{1000}n = O(n)$
$f_5(n) = n \log n = O(n.n^c) = O(n^{c+1})$
Vì f_3 giảm dần khi n tiến đến vô cùng nên f_3 bé nhất
$\rightarrow f_3 < f_4 < f_5 < f_1 = f_2$

Group 2

$$f_1(n) = 2^{2^{1000000}}$$
$$f_2(n) = 2^{100000n}$$
$$f_3(n) = \binom{n}{2}$$
$$f_4(n) = n\sqrt{n}$$

Ta có :
$f_1(n) = O(c)$
$f_2(n) = O(2^n)$
$f_3(n) = \frac{n(n-1)}{2} = O(n^2)$
$f_4(n) = n * n^{1/2} = n^{3/2} = O(n^{3/2})$

$\rightarrow f_1 < f_4 < f_3 < f_2$

Group 3

$$f_1(n) = n^{\sqrt{n}}$$
$$f_2(n) = 2^n$$
$$f_3(n) = n^{10}.2^{n/2}$$
$$f_4(n) = \sum_{i=1}^{n}(i+1)$$

Ta có:

$f_1(n) = 2^{\sqrt{n}.\log n} = 2^{n^{1/2}.\log n} = 2^{O(n^{1/2+c})}$ (với c rất nhỏ) (1)

$f_2(n) = 2^n = 2^{O(n)}$

$f_3(n) = n^{10}.2^{n/2} = \left(2^{\log n}\right)^{10}.2^{n/2} = 2^{10.\log n + n/2} = 2^{O(n)}$

$f_4(n) = \sum_{i=1}^{n}(i) + \sum_{i=1}^{n}(1)$

$= \frac{(n+1).n}{2} + n$

$= \frac{n^2}{2} + \frac{3n}{2} = \frac{1}{2}.(2^{log_n})^2 + \frac{3}{2}(2^{\log n})$

$= \frac{1}{2}.2^{2\log n} + \frac{3}{2}(2^{\log n}) = 2^{O(n^c)}$ (với c rất nhỏ)

Với c là một hằng số rất nhỏ, ta thấy: $n > n^{1/2+c} > n^c$ vậy nên $2^n > 2^{1/2+c} > 2^c$.

Vậy ta được $f_3(n) > f_1(n) > f_4(n)$

Mà $f_2(n) = 2^{O(n)}$ và $f_3(n) = 2^{O(n)}$, nên xét theo Big-O, ta được:

$$f_4(n) < f_1(n) < f_3(n) = f_2(n)$$

Xét hàm $y_1 = n$, khi lấy đạo hàm ta được $y_1' = 1$, vậy hàm số y_1 sẽ là hàm đồng biến và tăng với tốc độ không đổi là 1 hằng số. Trong khi đó hàm số $y_2 = 10.\log n + n/2$, khi đạo hàm ta sẽ được $y_2' = \frac{10}{n.\ln 10} + \frac{1}{2}$, ta thấy hàm y_2' là hàm đồng biến tăng với tốc độ phụ thuộc vào biến n, khi n càng lớn thì hàm tăng càng chậm. Từ đó suy ra khi n đủ lớn, thì $y_1 > y_2$ hay $2^n > 2^{10.\log n + n/2}$

Vậy $f_2(n) > f_3(n)$, nên:

$$f_4(n) < f_1(n) < f_3(n) < f_2(n)$$

Group 4

$$f_6(n) = n^{\sqrt{n}} = 2^{\sqrt{n}\log n}$$
$$f_7(n) = \pi^n = 2^{n\log\pi}$$
$$f_8(n) = 2^{n^4}$$
$$f_9(n) = n^{4logn} = 2^{4logn^2}$$

Ta có: $O(\sqrt{n}) > O(\log n)$

$\rightarrow O(\sqrt{n}\log n) > O(\log n^2)$

$\rightarrow O(2^{\sqrt{n}\log n}) > O(2^{4logn^2})$

$\rightarrow f_6 > f_9 \quad (1)$

Ta có: $O(\sqrt{n}) > O(\log n)$

$\rightarrow O(\sqrt{n}^2) > O(\sqrt{n}\log n)$

$\rightarrow O(2^{nlog\pi}) > O(2^{\sqrt{n}\log n})$

$\rightarrow f_7 > f_6 \quad (2)$

Ta có: $O(n^4) > O(n)$

$\rightarrow O(2^{n^4}) > O(2^{n\log\pi})$

$\rightarrow f_8 > f_7 \quad (3)$

Từ $(1), (2), (3) \Rightarrow f_9 < f_6 < f_7 < f_8$

Group 5

$$f_5(n) = n^n$$
$$f_7(n) = n\log n$$
$$f_8(n) = 2^{n/2}$$
$$f_9(n) = 3^{\sqrt{n}}$$
$$f_{10}(n) = 4^{n^{1/4}}$$

Ta có :
$f_6(n) = n^{\sqrt{n}} = n^{n^{\frac{1}{2}}} = 2^{n^{\frac{1}{2}}\log n} = 2^{O(n^c.n^{\frac{1}{2}})} = 2^{O(n^{c+\frac{1}{2}})}$
$f_7(n) = n\log n = 2^{\log n}.2^{\log(\log n)} = 2^{\log n+\log(\log n)} = 2^{O(\log n)}$
$f_8(n) = 2^{\frac{n}{2}} = 2^{O(n)}$
$f_9(n) = 3^{\sqrt{n}} = 3^{n^{\frac{1}{2}}} = 2^{\log 3.n^{\frac{1}{2}}} = 2^{O(n^{\frac{1}{2}})}$
$f_{10}(n) = 4^{n^{\frac{1}{4}}} = 2^{2n^{\frac{1}{4}}} = 2^{O(n^{\frac{1}{4}})}$

Vì $n^{\frac{1}{4}}$vẫn là hàm mũ của n nên độ phức tạp của nó sẽ vẫn lớn hơn logn khi n đủ lớn
$\rightarrow f_7(n) < f_{10}(n) < f_9(n) < f_6(n) < f_8(n)$

Group 6

$$f_1(n) = n^{0.999999} * log(n)$$
$$f_2(n) = 10000000n$$
$$f_3(n) = 1.000001^n$$
$$f_4(n) = n^2$$

Ta có :
$f_1(n) = n^{0.999999} * log(n) = O(n^{0.999999}.n^c) = (n^{c+0.999999})$ với c rất bé và c > 0

$f_2(n) = O(n)$
$f_3(n) = O(1.000001^n)$(hàm lũy thừa => tăng nhanh nhất)
$f_4(n) = O(n^2)$
Với c + 0.999999 < 1 thì ta có:
$\rightarrow f_1 < f_2 < f_4 < f_3$

Group 7

$$f_1(n) = n^\pi$$
$$f_2(n) = \pi^n$$
$$f_3(n) = \binom{n}{5}$$
$$f_4(n) = \sqrt{2^{\sqrt{n}}}$$
$$f_5(n) = \binom{n}{n-4}$$
$$f_6(n) = 2^{(\log n)^4}$$
$$f_7(n) = n^{5.(\log n)^2}$$
$$f_8(n) = n^4.\binom{n}{4}$$

Ta có: (với c là hằng số rất nhỏ, c « 1)

$$f_1(n) = O(n^\pi) \tag{1}$$
$$f_2(n) = \pi^{O(n)} \tag{2}$$
$$f_3(n) = \frac{n!}{5!.(n-5)!} = \frac{1}{5!}.[n.(n-1).(n-2).....(n-4)] = O(n^5) \tag{3}$$
$$f_4(n) = (2^{n^{1/2}})^{1/2} = 2^{\frac{n^{1/2}}{2}} = 2^{O(n^{\frac{1}{2}})} \tag{4}$$
$$f_5(n) = \binom{n}{n-4} = \frac{n!}{(n-4)!.4!} = \frac{1}{4!}.[n.(n-1).(n-2).(n-3)] = O(n^4) \tag{5}$$
$$f_6(n) = 2^{(\log n)^4} = 2^{O(n^{4c})} \tag{6}$$
$$f_7(n) = n^{5.(\log n)^2} = (2^{\log n})^{5.\log^2 n} = 2^{5.\log^3 n} = 2^{O(n^{3c})} \tag{7}$$
$$f_8(n) = n^4.\frac{n!}{4!.(n-4)!} = n^4.\frac{1}{4!}.[n.(n-1).(n-2).(n-3)] = O(n^8) \tag{8}$$

Từ (1), (3), (5), (8), ta có:

$$\pi < 4 < 5 < 8$$
$$\Leftrightarrow O(n^\pi) < O(n^4) < O(n^5) < O(n^8)$$
$$\Leftrightarrow f_1(n) < f_5(n) < f_3(n) < f_8(n)$$

Từ (4), (6), (7), ta có:

$$\begin{aligned} & 3c < 4c < \frac{1}{2} \\ & \Leftrightarrow n^{3c} < n^{4c} < n^{\frac{1}{2}} \\ & \Leftrightarrow O(n^{3c}) < O(n^{4c}) < O(n^{\frac{1}{2}}) \\ & \Leftrightarrow 2^{O(n^{3c})} < 2^{O(n^{4c})} < 2^{O(n^{\frac{1}{2}})} \\ & \Leftrightarrow f_7(n) < f_6(n) < f_4(n) \end{aligned}$$

Xét $f_8(n)$ là một hàm số với bậc cao nhất là n^8. Xét $n^8 = (2^{\log n})^8 = 2^{8.\log n} = 2^{O(n^c)}$. Vì $n^c < n^{3c}$ nên $f_8(n) < f_7(n)$, vậy, ta được:

$$f_1(n) < f_5(n) < f_3(n) < f_8(n) < f_7(n) < f_6(n) < f_4(n)$$

Xét: $n^{\frac{1}{2}} < n \Leftrightarrow 2^{n^{\frac{1}{2}}} < 2^n < \pi^n \Leftrightarrow 2^{O(n^{\frac{1}{2}})} < 2^{O(n)} < \pi^{O(n)}$. Vậy $f_4(n) < f_2(n)$, nên:

$$f_1(n) < f_5(n) < f_3(n) < f_8(n) < f_7(n) < f_6(n) < f_4(n) < f_2(n)$$

Câu 3: Chứng minh dùng định nghĩa của ký hiệu tiệm cận(không dùng lim)

3.a

$$n^4 + n + 1 \notin O(n^2)$$

Giả sử $n^4 + n + 1 \in O(n^2)$

Xét $C \leq 1$:

$\Rightarrow n^4 \geq Cn^2 \quad \forall n > 1$

$\Rightarrow n^4 + n + 1 \leq Cn^2$ sai $\forall n$

Xét $C > 1$:

Nếu $1 < n_0 \leq C$ chọn $n = C$

Ta được: $C^4 + C + 1 \leq C^3 \quad (1)$

mà $C^4 \geq C^3$

$\Rightarrow (1)$ sai

Nếu $n_0 > C$ chọn $n = kC, k \in N^*$

Ta được: $k^4C^4 + kC + 1 \leq k^2C^3 \quad (2)$

mà $k^4C^4 \geq k^2C^3$

$\Rightarrow (2) sai$

Vậy không tồn tại C, n_0 để thỏa $n^4 + n + 1 \leq Cn^2 \quad \forall n > n_0$

$\Rightarrow n^4 + n + 1 \notin O(n^2)$

3.b

$$\text{O(C.f(n)) = O(f(n)) với C là hằng số}$$

Xét 1 hàm g(n) bất kỳ , g(n) $\in$ O(C.f(n))
$\rightarrow \exists c_1 \in \mathbb{R}_+; n_1 \in \mathbb{N}$ sao cho :
$g(n) \leq c_1.C.f(n), \forall n \geq n_1$
Chọn $c_2 = c_1.C, n_2 = n_1$
$\rightarrow \exists c_2 \in \mathbb{R}_+, n_2 \in \mathbb{N}$ sao cho:
$g(n) \leq c_2.f(n), \forall n \geq n_2$
$\rightarrow g(n) \in O(f(n))$

Xét 1 hàm g(n) bất kỳ , g(n) $\in$ O(f(n))
$\rightarrow \exists c_1 \in \mathbb{R}_+; n_1 \in \mathbb{N}$ sao cho :
$g(n) \leq c_1.f(n), \forall n \geq n_1$
Chọn $c_2 = \frac{c_1}{C}, n_2 = n_1$
$\rightarrow \exists c_2 \in \mathbb{R}_+, n_2 \in \mathbb{N}$ sao cho:
$g(n) \leq c_2.C.f(n), \forall n \geq n_2$
$\rightarrow g(n) \in O(C.f(n))$
Vậy O(C.f(n)) = O(f(n))

3.c

Chứng minh nếu (f(n) = O(g(n))) và (g(n) = O(h(n))), thì (f(n) = O(h(n)))
Giả sử (f(n) = O(g(n))) và (g(n) = O(h(n))).
Ta có:

$$f(n) = O(g(n)) => \exists f(n) \leq C_0 \cdot g(n) \forall (n > n_0).(1)$$
$$g(n) = O(h(n)) => \exists g(n) \leq C_1 \cdot h(n) \forall (n > n_0).(2)$$

Để $f(n) = O(h(n))$
$\exists C_2, n_2 : f(n) \leq C_2 * h(n) \forall (n > n_2)$.

Chọn $(n_2 = \max(n_0, n_1))$ và $(C_2 = C_0 \cdot C_1)$. Khi đó:

$f(n) \leq C_0 \cdot g(n)$ (theo (1)) $\leq C_0 \cdot (C_1 \cdot h(n))$ (theo (2)) $= (C_0 \cdot C_1) \cdot h(n) = C_2 \cdot h(n) \forall (n > n_2)$

Hay $f(n) = O(h(n))$.
Như vậy, ta đã chứng minh rằng nếu (f(n) = O(g(n))) và (g(n) = O(h(n))), thì (f(n) = O(h(n)))

3.e

$$g(n) \in O(h(n)) \Rightarrow O(g(n)) \subseteq O(h(n))$$

Lấy $f(n)$ bất kỳ thuộc $O(g(n)$

Ta có được:

$\exists C_1 \in R^+, n_1 \in N^* :$

$f(n) \leq C_1 g(n)\ \forall n \geq n_1\ (1)$

Ta có: $g(n) \in O(h(n))$

$\Rightarrow \exists C_2 \in R^+, n_2 \in N^* :$

$g(n) \leq C_2 h(n)\ \forall n \geq n_2\ (2)$

Từ $(1), (2)$ ta được:

$\exists C_1, C_2 \in R^+, n_1, n_2 \in N^* :$

$f(x) \leq C_1 C_2 h(n)\ \forall n \geq n_1 + n_2$

Vậy một thành phần bất kì nào từ $O(g(n))$ đều thuộc O(h(n))

$\rightarrow g(n) \in O(h(n)) \Rightarrow O(g(n)) \subseteq O(h(n))$

3.f

$$\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$$

Xét 1 hàm f(n) bất kỳ , f(n) $\in \Theta(g(n))$
$\rightarrow \exists c_1, c_2 \in \mathbb{R}_+; n_1 \in \mathbb{N}$ sao cho :
$c_1 g(n) \leq f(n) \leq c_2 g(n), \forall n \geq n_1$
Chọn $d_1 = c_1, d_2 = c_2, n_2 = n_1$
$\rightarrow \exists d_1, d_2 \in \mathbb{R}_+, n_2 \in \mathbb{N}$ sao cho:
$\begin{cases} d_1 g(n) \leq f(n), (\text{thỏa} f(n) = \Omega(g(n))) \\ f(n) \leq d_2 g(n), (\text{thỏa} f(n) = O(g(n))) \end{cases}$
$\rightarrow f(n) = O(g(n)) \cap \Omega(g(n)) \rightarrow g(n) \in O(f(n))$

Xét 1 hàm f(n) bất kỳ , $f(n) \in \Omega(g(n)), f(n) \in O(g(n))$
$\rightarrow \exists c_1, c_2 \in \mathbb{R}_+; n_1 \in \mathbb{N}$ sao cho :
$\begin{cases} c_1 g(n) \leq f(n), (\text{thỏa} f(n) = \Omega(g(n))) \\ f(n) \leq c_2 g(n), (\text{thỏa} f(n) = O(g(n))) \forall n \geq n_1 \end{cases}$

Chọn $d_1 = c_1, d_2 = c_2, n_2 = n_1$
$\rightarrow \exists d_1, d_2 \in \mathbb{R}_+, n_2 \in \mathbb{N}$ sao cho:
$d_1 g(n) \leq f(n) \leq d_2 g(n), \forall n \geq n_2$
$\rightarrow f(n) \in \Theta(g(n))$
Vậy $\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$

3.g

Đề bài: $n + n^2 * O(\ln n) = O(n^2 \ln n)$
Vế 1 CM: $n^2 O(ln(n)) \subseteq O(n^2 \ln n)$
Xét 1 phần tử g(n) bất kỳ thuộc O(ln(n))
$\exists$ n_0 và a $\in R^+$ sao cho:
$g(n) \leq aln(n) \forall n > n_0$
Để $n^2 g(n) \in O(n^2 ln(n))$
Thì $n^2 g(n) \leq bn^2 ln(n) \forall n > n_1 (1)$
Luôn tồn tại b = a, $n_1 = n_0$ thỏa (1)
Hay mọi phần tử của $n^2 O(ln(n)) \in O(n^2 \ln n)$

Vế 2 CM: $O(n^2 \ln n) \subseteq n^2 O(ln(n))$
Xét 1 phần tử g(n) bất kỳ thuộc $O(n^2 ln(n))$
$\exists n_0$ và a $\in R^+$ sao cho:
$g(n) \leq an^2 ln(n) \forall n > n_0$
Để $g(n) \in n^2 O(ln(n))$
Thì $g(n) \leq bn^2 ln(n) \forall n > n_1 (2)$
Luôn tồn tại b = a, $n_1 = n_0$ thỏa (2)
Hay mọi phần tử của $O(n^2 ln(n)) \in n^2 O(\ln n)$
$=> O(n^2 \ln n) \subseteq n^2 O(ln(n))$
Vậy $n + n^2 * O(\ln n) = O(n^2 \ln n)$

Câu 4: Các khẳng định bên dưới là đúng hay sai, vì sao ?

4.a/ Nếu $f(n) = \Theta(g(n))$ và $g(n) = \Theta(h(n))$, thì $h(n) = \Theta(f(n))$

Vì $f(n) = \Theta(g(n))$, nên $\exists c_1, c_2 \in R+, n_0 \in N$, sao cho:

$$c_1.g(n) \leq f(n) \leq c_2.g(n) \qquad \forall n \geq n_0$$

Vì $g(n) = \Theta(h(n))$, nên $\exists c_3, c_4 \in R+, n_1 \in N$, sao cho:

$$c_3.h(n) \leq g(n) \leq c_4.h(n) \qquad \forall n \geq n_1$$

Xét:

$$\begin{aligned} c_3.h(n) &\leq g(n) \\ \Rightarrow c_3.c_1.h(n) &\leq c_1.g(n) \leq f(n) \\ \Rightarrow h(n) &\leq \frac{1}{c_3.c_1}.f(n) \qquad (1) \end{aligned}$$

Xét:

$$\begin{aligned} g(n) &\leq c_4.h(n) \\ \Rightarrow f(n) &\leq c_2.g(n) \leq c_4.c_2.h(n) \\ \Rightarrow \frac{1}{c_4.c_2} f(n) &\leq h(n) \qquad (2) \end{aligned}$$

Từ (1) và (2) ta được:

$$\frac{1}{c_4.c_2}f(n) \leq h(n) \leq \frac{1}{c_3.c_1}.f(n)$$

Chọn $c_5 = \frac{1}{c_4.c_2}, c_6 = \frac{1}{c_3.c_1}, n_2 = max(n_0, n_1)$, theo định nghĩa Big Θ, ta được:

$$c_5.f(n) \leq h(n) \leq c_6.f(n)$$
$$\Rightarrow h(n) = \Theta(f(n))$$

Vậy khẳng định của đề bài là đúng.

4.b

$$f(x) = O(g(n)), g(n) = O(f(n)) \Rightarrow f(n) = g(n)$$

Giả sử $g(n) = Cf(n)$ với $C \in R^+, C \neq 1$

Ta có: với $C_1 = \frac{1}{C}$:

$$f(n) \leq CC_1f(n), \forall n \geq 1$$
$$\rightarrow f(n) = O(Cf(n))$$
$$\rightarrow f(n) = O(g(n)) \quad (1)$$

Ta có: với $C_2 = C$:

$$Cf(n) \leq C_2f(n), \forall n \geq 1$$
$$\rightarrow Cf(n) = O(f(n))$$
$$\rightarrow g(n) = O(f(n)) \quad (2)$$

Ta có (1),(2) mà $f(n) \neq Cf(n)$

$\Rightarrow$ Khẳng định ban đầu là sai

4.c

$$f(n) + O(f(n)) = \Theta(f(n))$$

Xét g(n) bất kỳ , $g(n) \in O(f(n))$
$\rightarrow \exists c \in \mathbb{R}^+, n_0 \in \mathbb{N}$, sao cho :
$g(n) \leq c \cdot f(n)$
$g(n) + f(n) \leq c \cdot f(n) + f(n)$
$g(n) + f(n) \leq (c+1) \cdot f(n)$ (1)
Ta có : $f(n) + g(n) \geq f(n)$ (2)
Từ (1), (2) :
Chọn $d_1 = 1, d_2 = c + 1$
$\rightarrow \exists d_1, d_2 \in \mathbb{R}^+, n_0 \in \mathbb{N}$, sao cho :
$d_1 \cdot f(n) \leq f(n) + g(n) \leq d_2 \cdot f(n)$
Vậy $f(n) + O(f(n)) = \Theta(f(n))$

4.d/ $2^{10n} = O(2^n)$

$2^{10n} = O(2^n)$
Tồn tại một C $\in R^+$ sao cho
$2^{10n} \leq C \cdot 2^n$
=> Vô lý (C không tồn tại)
=> Khẳng định ban đầu là sai.

4.e/ $2^{n+10} = O(2^n)$

Xét: $2^{n+10} = 2^n.2^{10} \Rightarrow 2^n.2^{10} \leq c.2^n$
Chọn $c = 2^{10}, n_0 = 1$, vậy theo định nghĩa Big O, ta được:

$$2^{n+10} \leq c.2^n$$
$$\Rightarrow 2^{n+10} = O(2^n)$$

4.f

$$\log_a n = \Theta(\log_b n)$$

Giả sử $\log_a n = \Theta(\log_b n)$

Ta có: $\exists C_1, C_2 \in R^+, n_0 \in N^*$ sao cho:

$$C_1 \log_b n \leq \log_a n \leq C_2 \log_b n \quad n \geq n_0$$

$$C_1 \log_b n \leq \frac{\log_b n}{\log_b a} \leq C_2 \log_b n \quad n \geq n_0$$

$$C_1 \leq \frac{1}{\log_b a} \leq C_2 \quad (1)$$

Luôn tồn tại C_1, C_2 thỏa (1)

$\Rightarrow$ Khẳng định ban đầu là đúng

Câu 5:

5.1/

$$T(n) = 3T(\frac{n}{2}) + n^2$$

a = 3, b = 2 ($a \geq 1, b \geq 2$)
f(n) = $n^2 = \Theta(n^2)(d = 2)$
Áp dụng định lý Master (1)(case 1) :
$a < b^d (3 < 2^2)$
$\rightarrow T(n) = \Theta(n^2)$

5.2/

$$T(n) = 7T(\frac{n}{3}) + n^2$$

a = 7, b = 3 ($a \geq 1, b \geq 2$)
f(n) = $n^2 = \Theta(n^2)(d = 2)$
Áp dụng định lý Master (1) (case 1) :
$a < b^d(7 < 3^2)$
$\rightarrow T(n) = \Theta(n^2)$

$5.3/T(n) = 3.T(n/3) + \frac{n}{2}$

Theo định lý Master, ta có:

$$\begin{aligned} &a = 3 \geq 1 \\ &b = 3 \geq 2 \\ &f(n) = \frac{n}{2} = \Theta(n) \qquad (d = 1) \\ &\Rightarrow a = b^d \qquad (3 = 3^1) \end{aligned}$$

Vậy áp dụng định lý Master dạng 1, trường hợp 2, ta có:

$$\begin{aligned} T(n) &= \Theta(n^d . \log n) \\ &= \Theta(n . \log n) \end{aligned}$$

5.4/

$$T(n) = 16T(\frac{n}{4}) + n$$

$a = 16$

$b = 4$

$f(n) = n = \Theta(n^1)$

$a = 16 > 4^1 = b^1$

Theo định lý Master Dạng đơn giản - Case 3:

$T(n) = \Theta(n^{log_4 16}) = \Theta(n^2)$

5.5

$$T(n) = 2T(\frac{n}{4}) + n^{0.51}$$

a = 2, b = 4 ($a \geq 1, b\ e2$)
f(n) = $n^{0.51} = \Theta(n^{0.51})(d = 0.51)$
Áp dụng định lý Master (1) :
$a < b^d(2 < 4^{0.51})$
$\rightarrow T(n) = \Theta(n^{0.51})$

5.6/

$$T(n) = 3T(\frac{n}{2}) + n$$

a = 3, b = 2 $(a \geq 1, b \geq 2)$
f(n) = $n = \Theta(n)(d = 1)$
Áp dụng định lý Master (1)(case 3) :
$a > b^d(3 > 2)$
$\rightarrow T(n) = \Theta(n^{\log_2 3})$

5.7/ $T(n) = 3.T(n/3) + \sqrt{n}$

Ta có $T(n) = 3.T(n/3) + n^{1/2}$, nên:

$$\begin{aligned} & a = 3 \geq 1 \\ & b = 3 \geq 2 \\ & f(n) = n^{1/2} = \Theta(n^{1/2}) \qquad (d = 1/2) \\ & \Rightarrow a > b^d \qquad (3 > 3^{1/2}) \end{aligned}$$

Áp dụng định lý Master dạng 1 trường hợp 3, ta được:

$$\begin{aligned} T(n) &= \Theta(n^{\log_b a}) \\ &= \Theta(n) \end{aligned}$$

5.8/

$$T(n) = 4T(\frac{n}{2}) + cn$$

$a = 4$

$b = 2$

$f(n) = cn = \Theta(n^1)$

$a = 4 > 2^1 = b^1$

Theo định lý Master Dạng đơn giản - Case 3:

$T(n) = \Theta(n^{log_2 4}) = \Theta(n^2)$

5.9

$$T(n) = 4T(\frac{n}{4}) + 5n$$

a = 4, b = 4 $(a \geq 1, b\ e2)$
f(n) = $5n = \Theta(n)(d = 1)$
Áp dụng định lý Master (1) :
$a = b^d(4 = 4^1)$
$\rightarrow T(n) = \Theta(n \log n)$

5.10/

$$T(n) = 5T(\frac{n}{4}) + 4n$$

a = 5, b = 4 ($a \geq 1, b \geq 2$)
f(n) = $4n = \Theta(n)(d = 1)$
Áp dụng định lý Master (1)(case 3) :
$a < b^d(5 < 4)$
$\rightarrow T(n) = \Theta(n^{\log_4 5})$

5.11/ $T(n) = 4.T(n/5) + 5n$

Ta có:

$$\begin{aligned} &a = 4 \geq 1 \\ &b = 5 \geq 2 \\ &f(n) = 5n = \Theta(n) \qquad (d = 1) \\ &\Rightarrow a < b^d \qquad (4 < 5^1) \end{aligned}$$

Áp dụng Định lý Master dạng 1, trường hợp 1, ta có:

$$\begin{aligned} T(n) &= \Theta(n^d) \\ &= \Theta(n) \end{aligned}$$

5.12.

$$T(n) = 25T(\frac{n}{5}) + n^2$$

$a = 25$

$b = 5$

$f(n) = n^2 = \Theta(n^2)$

$a = 25 = 5^2 = b^2$

Theo định lý Master Dạng đơn giản - Case 2:

$T(n) = \Theta(n^2 \log n)$

5.13.

$$T(n) = 10T(\frac{n}{3}) + 17n^{1.2}$$

a = 10, b = 3 ($a \geq 1, b \geq 2$)
f(n) = $17n^{1.2} = \Theta(n^{1.2})(d = 1.2)$
Áp dụng định lý Master (1) :
$a > b^d(10 > 3^{1.2})$
$\rightarrow T(n) = \Theta(n^{\log_3 10})$

5.14/

$$T(n) = 7T(\frac{n}{2}) + n^3$$

a = 7, b = 2 ($a \geq 1, b \geq 2$)
f(n) = $n^3 = \Theta(n^3)(d = 3)$
Áp dụng định lý Master (1) (case 3):
$a < b^d(7 < 2^3)$
$\rightarrow T(n) = \Theta(n^3)$

5.15/ $T(n) = 4.T(n/2) + \log n$

Ta có:

$$\begin{aligned} &a = 4 \geq 1 \\ &b = 2 \geq 2 \\ &f(n) = \log n \in O(\log n) \in O(n) \in O(n^{\log_b a - \epsilon}) \\ &\Rightarrow \epsilon = 2 \end{aligned}$$

Áp dụng định lý Master dạng 2 trường hợp 1, ta được:

$$T(n) = \Theta(n^{\log_b a}) = \Theta(n^2)$$

5.16.

$$T(n) = 4T(\frac{n}{5}) + \log n$$

$a = 4$

$b = 5$

$f(n) = \log n$

Ta có: $f(n) = \log n = O(n^{\log_b a - 1}) = O(n)$

Theo định lý Master Dạng tổng quát - Case 1 :

$T(n) = \Theta(n^{log_b a}) = \Theta(n^{log_4 5})$

5.17.

$$T(n) = \sqrt{2}T(\frac{n}{2}) + \log n$$

a = $\sqrt{2}$, b = 2 , $\log_b a = \log_2 \sqrt{2} = \frac{1}{2}$
f(n) = $\log n = O(n^{\frac{1}{2} - \epsilon})$
Áp dụng định lý Master (2) , TH 1:
$\rightarrow T(n) = \Theta(n^{\log_b a}) = \Theta(n^{\frac{1}{2}})$

5.18/ $T(n) = 2.T(n/3) + n.\log n$

Ta có:(Định lý master dạng 2 và case 3)

$$\begin{aligned}
&a = 2 \geq 1 \\
&b = 3 \geq 2 \\
&f(n) = n.\log n \geq c.n \\
&\Rightarrow f(n) = \Omega(n) = \Omega(n^{\log_b a + \epsilon}) \\
&\Rightarrow \epsilon = 1 - \log_b a = 1 - \log_4 3 > 0
\end{aligned}$$

Xét điều kiện : $a.f(n/b) \leq c.f(n)$ với c < 1 và n đủ lớn.

$$\begin{aligned}
2.\left(\frac{n}{3}.\log\frac{n}{3}\right) &\leq c.(n.\log n) \\
\frac{2}{3}.n.\log\frac{n}{3} &\leq c.(n.\log n) \\
\frac{2}{3}.n.\log n - \frac{2}{3}.n.\log 3 &\leq c.(n.\log n)
\end{aligned}$$

Chọn c $= \frac{2}{3}$, ta thấy:

$$\begin{aligned}
\frac{2}{3}.n.\log n &\leq \frac{2}{3}.(n.\log n) \qquad \forall n > 1 \\
\frac{2}{3}.n.\log n - \frac{2}{3}.n.\log 3 &\leq \frac{2}{3}.(n.\log n) \qquad \forall n > 1 \\
\Rightarrow \frac{2}{3}.n.\log n - \frac{2}{3}.n.\log 3 &\leq \frac{2}{3}.n.\log n \forall n > 1
\end{aligned}$$

Vậy điều kiện đã thỏa, nên:

$$T(n) = \Theta(n\log n)$$

5.19/ $T(n) = 3.T(n/4) + n.\log n$

Áp dụng định lý Master (2) case 3, ta có:

$$\begin{aligned}
&a = 3 \geq 1 \\
&b = 4 \geq 2 \\
&f(n) = n.\log n \geq c.n \\
&\Rightarrow f(n) = \Omega(n) = \Omega(n^{\log_b a + \epsilon}) \\
&\Rightarrow \epsilon = 1 - \log_b a = 1 - \log_4 3 > 0
\end{aligned}$$

Xét điều kiện : $a.f(n/b) \leq c.f(n)$ với c < 1 và n đủ lớn.

$$\begin{aligned}
3.\left(\frac{n}{4}.\log\frac{n}{4}\right) &\leq c.(n.\log n) \\
\frac{3}{4}.n.\log n - \frac{3}{4}.n.\log 4 &\leq c.(n.\log n)
\end{aligned}$$

Chọn c = 0.9, ta thấy:

$$\frac{3}{4}n\log n \leq 0.9n\log n \qquad \forall n > 1$$
$$\Rightarrow \frac{3}{4}n\log n - \frac{3}{4}.n.\log 4 \leq 0.9n\log n \qquad \forall n > 1$$

Vậy điều kiện đã thỏa, nên:

$$T(n) = \Theta(n\log n)$$

5.20.

$$T(n) = 6T(\frac{n}{3}) + n^2\log n$$

$a = 6$

$b = 3$

$f(n) = n^2\log n$

$f(n) = \Omega(n^{\log_b a+3}) = \Omega(n^2)$

và $af(\frac{n}{b}) \leq cf(n)$

vì $6\frac{n^2}{9}\log\frac{n}{3} \leq cn^2\log n \quad (c = \frac{2}{3})$

$\Rightarrow \frac{2}{3}n^2(\log n - log3) \leq \frac{2}{3}n^2\log n$

Theo định lý Master Dạng tổng quát - Case 3:

$T(n) = \Theta(f(n)) = \Theta(n^2\log n)$

5.21.

$$T(n) = 3T(\frac{n}{5}) + \log^2 n$$

a = 3, b = 5 , $\log_b a = \log_5 3$
f(n) = $\log^2 n = O(n^{\log_5 3-\epsilon})$
Áp dụng định lý Master (2) , TH 1:
Vậy $T(n) = \Theta(n^{\log_5 3})$

5.22/ $T(n) = 2.T(n/2) + \frac{n}{\log n}$

Ta có:

$$a = 2 \geq 1$$
$$b = 2 \geq 2$$
$$f(n) = \frac{n}{\log n}$$

Định lý Master áp dụng cho các phương trình có dạng T(n) = $aT(\frac{n}{b}) + f(n))$ với (f(n) là một hàm $(O(n^c)), (c > -\infty)$. Tuy nhiên $f(n) = \frac{n}{\log n}$ không phải là một hàm $O(n^c)$ với bất kỳ giá trị c nào. Do đó, Định lý Master không thể áp dụng được cho phương trình đệ quy này.

5.23/ $T(n) = 2^n.T(n/2) + n^n$

Ta có:

$$a = 2^n \geq 1$$
$$b = 2 \geq 2$$
$$f(n) = n^n$$
$$\Rightarrow n^{\log_b a} = n^n$$

Do a không phải là một hằng số, nên không thể dùng định lý Master cho phương trình đệ quy trên.

5.24.

$$T(n) = 0.5T(\frac{n}{2}) + n$$

$a = 0.5 < 1$

⇒ Không thể sử dụng được định lý Master

5.25.

$$T(n) = T(\frac{n}{2}) + n(2 - cosn)$$

a = 1, b = 2 , $\log_b a = \log_2 1 = 0$
f(n) = $n(2 - cosn) = \Omega(n^{0+1})$
f(n) = $\Theta(n)$ vì khi n tăng lên thì giá trị của cosn chỉ từ -1 đến 1 nên khi giá trị của n đủ lớn thì độ phức tạp của f(n) chỉ còn phụ thuộc vào n
Áp dụng định lý Master (2) TH 3:

$$\frac{n}{2}(2 - cos\frac{n}{2}) \leq c \cdot n(2 - cosn)$$
$$c \geq \frac{2 - cos\frac{n}{2}}{4 - 2cosn}(1)$$

Ta có :
$\begin{cases} 1 \leq 2 - cos\frac{n}{2} \leq 3 \\ 2 \leq 4 - 2cosn \leq 6 \end{cases}$

$\rightarrow \frac{2-cos\frac{n}{2}}{4-2cosn} \leq \frac{3}{2}(2)$

Từ (1), (2) :
$c \geq \frac{3}{2}$ (vi phạm)
Vậy không thể dùng định lý Master để giải bài này.

5.26

$$T(n) = 64T(\frac{n}{8}) - n^2 \log n$$

$a = 64 \geq 1$
$b = 8 \geq 2$
$f(n) = -n^2 \log n$ là một hàm âm

Định lý Master không thể áp dụng được cho phương trình đệ quy này.

5.27/ $T(n) = T(n/2) + 2^n$

Áp dụng định lý Master(2) case 3, ta có:

$a = 1 \geq 1$
$b = 2 \geq 2$
$f(n) = 2^n$
$\Rightarrow n^{\log_b a} = n^{\log_2 1} = 1$

Vậy ta có:

$$f(n) = 2^n = \Omega(n^{\log_2 1+\epsilon}) \qquad \forall \epsilon > 0$$

Và thỏa điều kiện:

$a.f(n/b) \leq c.f(n)$
$\Leftrightarrow 2^{n/2} \leq c.2^n \qquad$ (với c = 1)

Vì thế: $T(n) = \Theta(2^n)$

5.28.

$$T(n) = 16T(\frac{n}{4}) + n!$$

$a = 16$

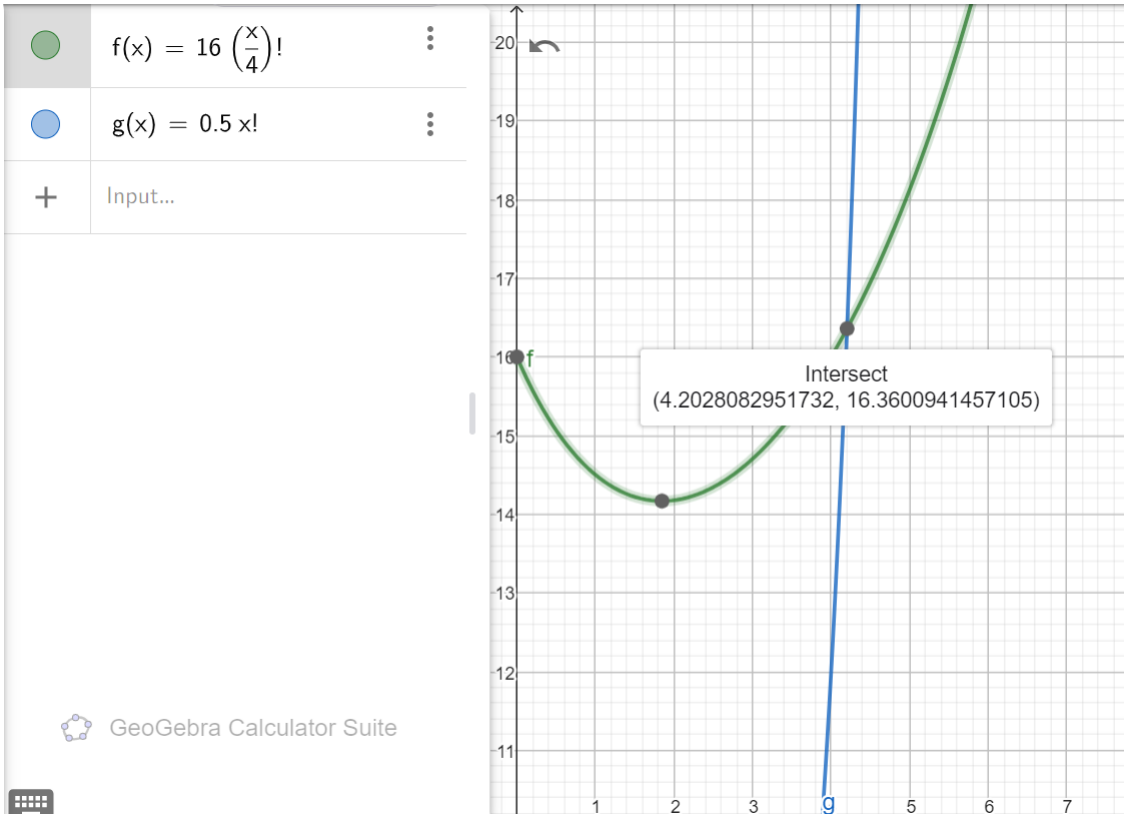
$b = 4$

$f(n) = n!$

$\rightarrow f(n) = n! = \Omega(n^{\log_b a+\epsilon}) = \Omega(n^{2+\epsilon}) \quad \forall \epsilon > 0$

và $af(\frac{n}{b}) \leq cf(n)$

Chọn C = 0,5: $16\frac{n}{4}! \leq 0,5 * n! \forall n \geq 5$



Theo định lý Master Dạng tổng quát - Case 3:

$T(n) = \Theta(f(n)) = \Theta(n!)$