انجمن علمی مهندسی کامپیوتر
دانشگاه مازندران

به نام خدا

تمرین اول ساختمان داده‌ها و الگوریتم

الگوریتم‌ها و حل مسئله - پیچیدگی زمانی - الگوریتم های بازگشتی

استاد:

دکتر روستایی

استادیار:

عرشیا عموزاد

۲۴ مهر ۱۴۰۴

۱. توابع زیر را از نظر رشد با هم مقایسه کنید.

$g(n) = n^{\log n}$
$f(n) = (logn)^n(n)$

۲. کدام یک از گزاره های زیر درباره‌ی توابع ذکر شده صحیح است؟

$f(n) = 4^{\log n}$
$g(n) = (\log n)^{\log n}$
$h(n) = \log^2 n$

$f(n) = O(g(n), g(n) = \Omega(h(n)))$ ◯
$g(n) = \Omega(h(n), h(n) = \Omega(f(n)))$ ◯
$f(n) = O(h(n), g(n) = \Omega(f(n)))$ ◯
$h(n) = O(g(n), f(n) = \Theta(g(n)))$ ◯

۳. توابع زیر را از نظر نرخ رشد به ترتیب صعودی مرتب کنید.

$f_5(n) = n^{\log\log n}$ ،$f_4(n) = n^{\log n}$ ،$f_3(n) = (\log n)^{\log n}$ ،$f_2(n) = 2^{\sqrt{n}}$ ،$f_1(n) = n^{1.01}$

۴. در یک زمستان سرد، خرس قطبی n قطعه گوشت دقیقا به اندازه‌های ۱، ۲ تا n را در غاری ذخیره کرده است. او هر روز یکی از این قطعه گوشت ها را به صورت تصادفی انتخاب می‌کند. اگر اندازه‌ی گوشت عدد فردی بود، آن را کاملا می‌خورد. اگر زوج بود، آن را دقیقا نصف می‌کند، یک نصف آن را می‌خورد و نصف دیگر را مجددا در غار قرار می‌دهد. اگر گوشتی موجود نباشد خرس می‌میرد. با این الگوریتم، برای n های خیلی بزرگ روزهای باقیمانده از عمر خرس قطبی تابع کدام یک از گزینه های خواهد بود؟

$O(n^2)$ ◯ $O(n\log n)$ ◯ $O(\log n)$ ◯ $O(n)$ ◯

۵. پیچیدگی زمانی توابع زیر را پیدا کنید.

۱. $T(n) = 4T(\frac{n}{2}) + \log^2 n!$
۲. $f(n) = \log(n!)$

۶. درباره‌ی اجرای الگوریتم مرتب سازی ادغامی (Merge Sort) بر روی آرایه‌ای به طول n به سوالات زیر پاسخ دهید.

۱. اگر در مرحله‌ی تقسیم، الگوریتم به جای نصف شدن دقیق به نسبت های دلخواه مانند $\frac{1}{3}$ و $\frac{2}{3}$ یا $\frac{1}{4}$ و $\frac{3}{4}$ تقسیم شود پیچیدگی زمانی الگوریتم چگونه تغییر می‌کند؟

۲. فرض کنید همه‌ی عناصر آرایه از قبل بصورت صعودی مرتب شده باشند. در این صورت تعداد مقایسه های الگوریتم نسبت به حالت تصادفی چه تغییری می‌کند؟

۷. با اجرای برنامه‌ی زیر چه عددی چاپ می‌شود؟

```c
#include <stdio.h>

int func(int, int);

void main() {
    int n = 2;
    int m = 3;
    printf("%d\n", func(n, m));
}

int func(int m, int n) {
    if (m == 0) return n + 1;
    if (n == 0) return func(m - 1, 1);
    return func(m - 1, func(m, n - 1));
}
```