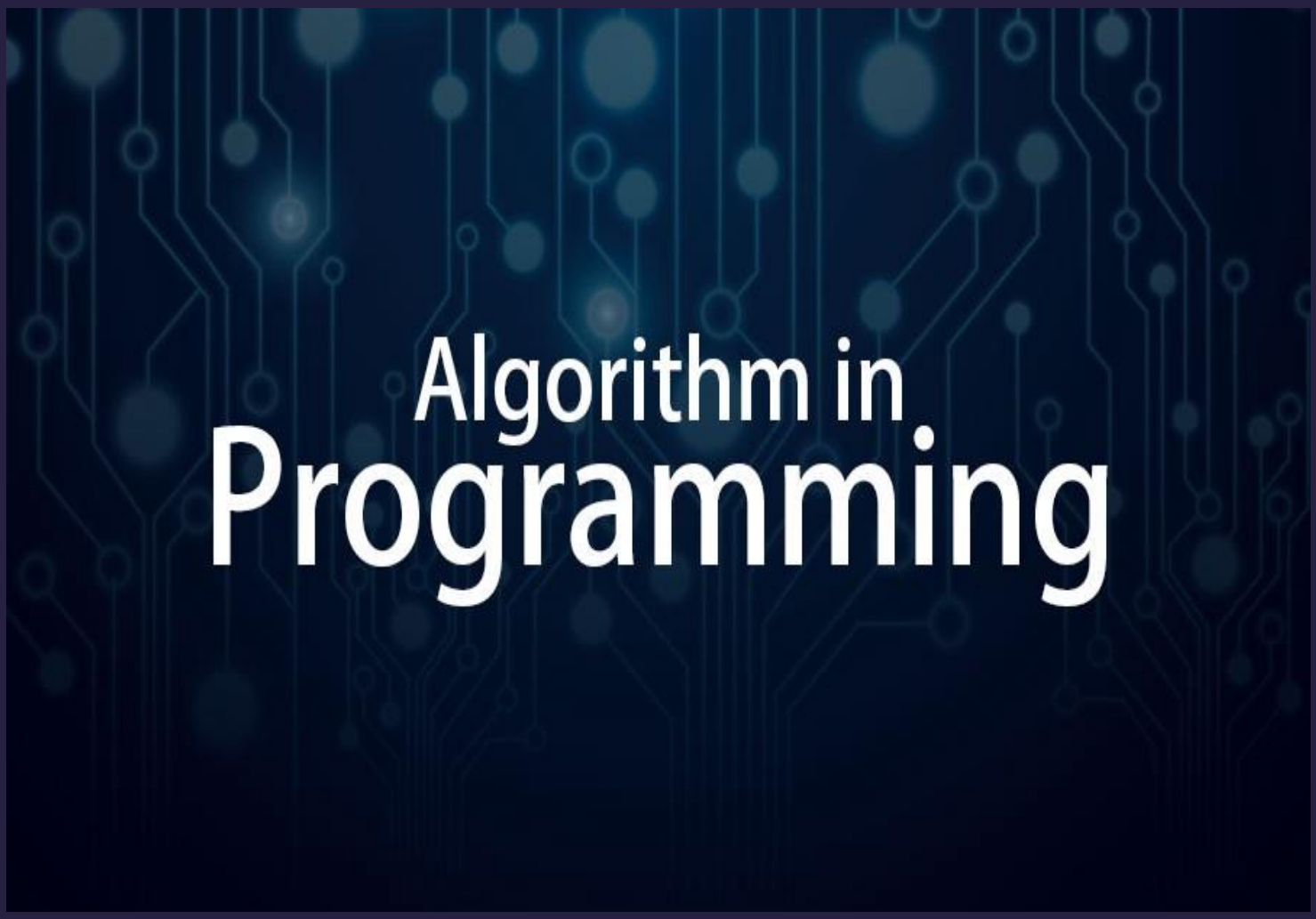


تحلیل و طراحی الگوریتم ها

تمرین سری اول

مصطفی فضلی 9822803

استاد مرضیه رحیمی 1400/01/17

دانشگاه صنعتی شاهرود

تحلیل و طراحی الگوریتم ها

تمرین سری اول

1 . for each function f(n) and time t in the following table, determine the largest size n of a problem that can be solved in time t, assuming that the algorithm to solve the problem takes f(n) microseconds.

	1 second	1 minute	1 hour	1 day	1 month	1 year	1 century
Lg n	2^{10^6}	$2^{6\times10^7}$	$2^{36\times10^8}$	$2^{864\times10^8}$	$2^{2592\times10^9}$	$2^{94608\times10^{10}}$	$2^{94608\times10^{12}}$
$\sqrt{n}$	10^{12}	36×10^{14}	1296×10^{16}	746496×10^{16}	6718464×10^{18}	895067×10^{24}	895067×10^{28}
n	10^6	6×10^7	36×10^8	864×10^8	2592×10^9	94608×10^{10}	94608×10^{12}
n lg n	62746	2801417	133378058	2755147513	71870856404	797633893349	68654697441062
n^2	1000	24494897	60000	293940	1609970	30758410	307584135
n^3	100	390	1530	4420	13735	98170	455660
2^n	20	25	30	36	42	49	56
$n!$	10	11	12	13	15	17	18

2 . find the following codes' time complexity.
*** just write the time complexity like this : o(n2)**

A)

```
for (i = o; i <= n - 1; i++)
    for (j = 1; j <= n - i; j++)
        a[j][i + j] = i;
```

⇨ $0(n^2)$ $\sum_{i=0}^{n-1} \sum_{j=1}^{n-1}(t)$

B)

```
for (i = 1; i <= n; i++)
    for (j = 1; j <= i; j++)
        for (k = 1; k <= n; k++)
            x = x + 1;
```

⇨ $0(n^3)$

تحلیل و طراحی الگوریتم ها

تمرین سری اول

C)

```
for (i = 1; i <= n; i++){
    for (i = 1; i <= n; i++)
        k = k + 1; j = 1;
    while (j < n){
        k = k + 1; j = 2 * j; }
    }
```

⇨ $0\big(n(n+\log n)\big)$

D)

```
i = n;
while( i >= 1){
    for (j = 1; j <= n; j++)
        x = x + 1;
        i = i / 3;
}
```

⇨ $O(n(log_3 n))$

E)

```
for (int i = n; i > 1; i = sqrt(i))
    { // some O(1) expressions }
```

⇨ $O(log\, n)$

F)

```
for (int i=1; i<=n; i++) {
    int p = pow(i, k);
```

⇨ O (n × n^k)

تحلیل و طراحی الگوریتم ها

تمرین سری اول

3 . You know that in the worst case, Merge Sort works better than insertion sort. But small problems of insertion sort can be faster in practice than the Merge Sort, so in practice It can be possible to improve the time complexity (running time) by solving sub-problems in the Merge Sort by Insertion Sort suppose that we want to organize n k Small sub-problems with length k in Merge Sort with Insertion Sort, explain the related algorithm and calculate It's time complexity. (OK?)

توضیح : ترکیب مرتب سازی درجی و ادغام برای بهبود زمان اجرا.

این ترکیب یکی از ترکیب های معروف الگوریتم به اسم مرتب سازی ادغامی-درجی است که توسط آقای فورد و آقای جانسون انتشار یافت، این الگوریتم در مسابقات برنامه نویسی نیز استفاده میشد اما امروزه الگوریتم هایی هستند که با همین تعداد مقایسه، زمان کمتری برای مرتب سازی مصرف میکنند. روش کار این الگوریتم بدین صورت است:
مرتب سازی ادغامی حافظه زیادی نیاز دارد و زمان آن نیز کمی بیشتر از ادغامی است، اما مرتب سازی درجی بدون استفاده از حافظه کمیکی مقایسه را انجام میدهد.
می توان این را بسته به طریقه پیاده سازی کاربر تعریف کرد اما اگر به طور معمول و میانگین در نظر بگیریم، می توان گفت که برای هنگامی که سایز آرایه ما کمتر از 100 باشد بهتر است از الگوریتم درجی و برای بیشتر از 100 از الگوریتم ادغامی استفاده کنیم.

منبع: استک اورفلو و سایت مغز باز و ویکی پدیا

4 . Reverse : the array of A[1…n] is an array of n different element. We call (i , j) a reverse if :
for any (i , j) => i < j and a[i] > a[j]

Find 5 Reverses of the following array :
[2,3,8,6,1]
Place of element [1,2,3,4,5]

Answer :
If starts from left =>
{2,1} - {3,1} - {8,6} - {8,1} - {6,1}

تحلیل و طراحی الگوریتم ها
تمرین سری اول

5 . A. Explain all four steps of Heap Sort Algorithm coming below . (at most on one page)

1. Build-MaxHeap(A)
2. for length (A) down to 2
3. do exchange A[1] A[i]
4. Max-heapify (A , 1 , i – 1)

به صورت ساده میتوان بیان کرد :
در ابتدا هرم بییشینه را تشکیل میدهیم . درخت بیشینه بدین صورت تشکیل می شود که ابتدا از بالاترین و چپ ترین نود شروع میکنیم و یکی یکی عناصر را درج می کنیم.
با یک حلقه تمامی عناصر تا تا عنصر اول پیمایش میکنیم، در این پیمایش بررسی میکنیم که آیا نیاز به جابجایی عنصر با با عنصر دیگری است یا خیر، اگر نیاز به جابجایی بود، عملیات Heapify را برای ان عنصر انجام میدهیم . (همچنین میتوانیم پیمایش حلقه را تا نصف طول آرایه نیز انجام بدهیم)
این کار را آنقدر انجام میدهیم تا همه عناصر از قانون بزرگتر بودن نود پدر از نود های فرزند پیروی کنند!

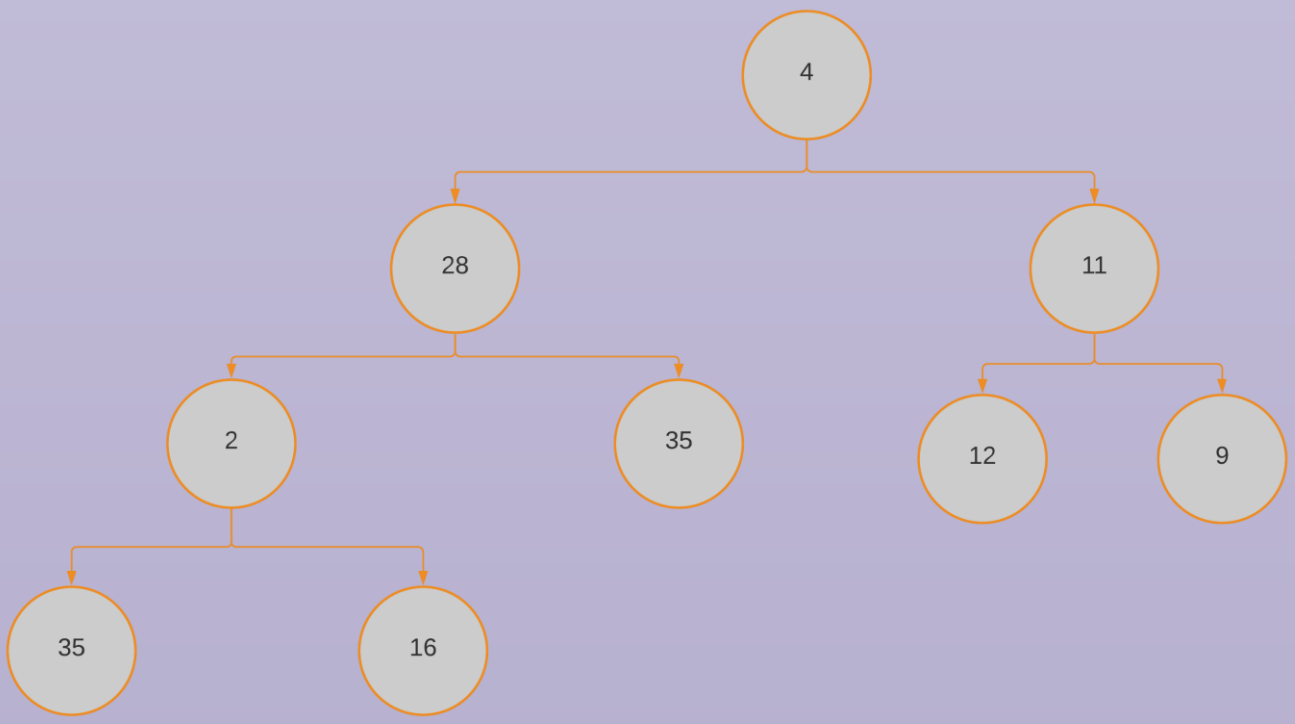
B) Please illustrate the operation of Heap Sort on the following Array :

4	28	11	2	35	12	9	35	16
4	28	11	35	35	12	9	2	16
4	35	11	28	35	12	9	2	16
35	4	11	28	35	12	9	2	16
35	35	11	28	4	12	9	2	16
35	35	12	28	4	11	9	2	16

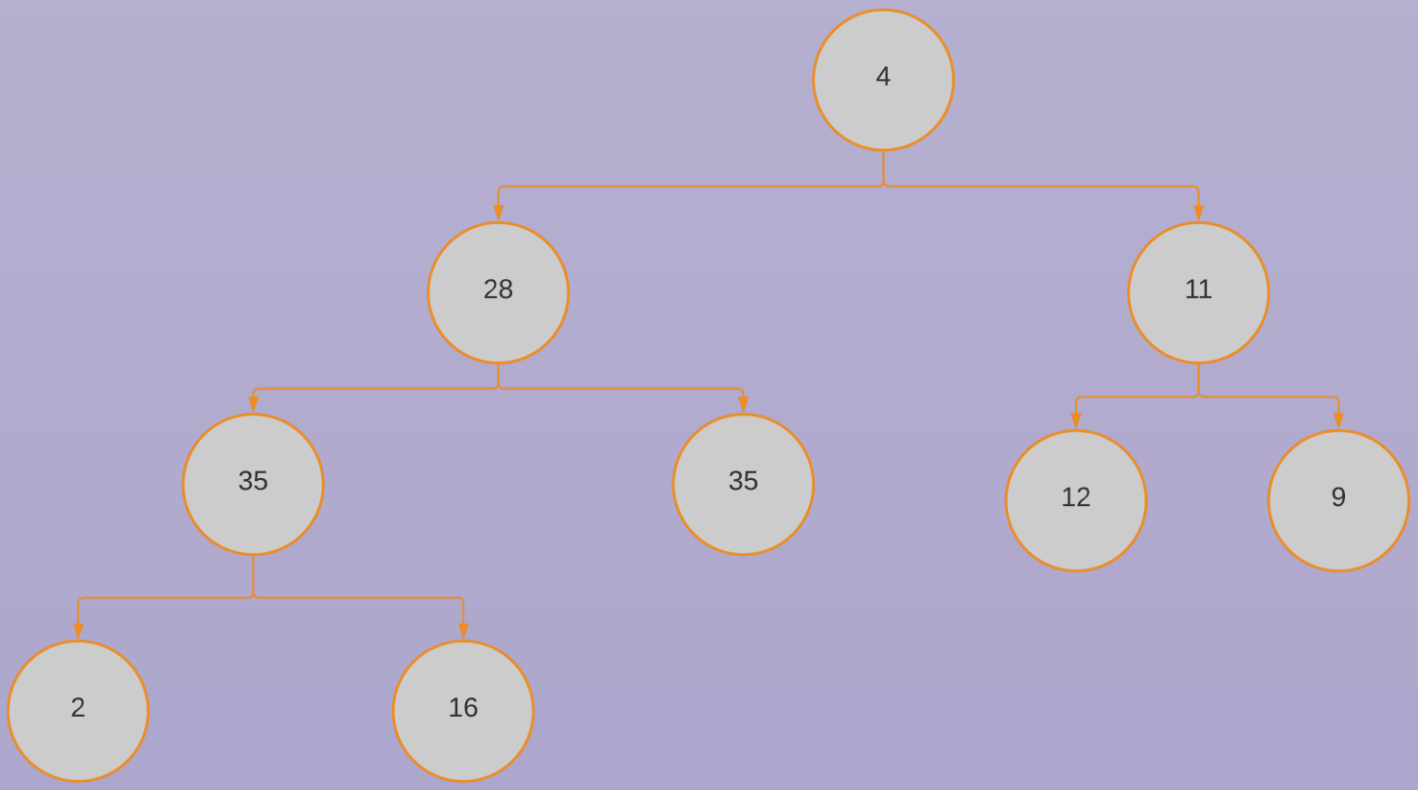
تحلیل و طراحی الگوریتم ها

تمرین سری اول

.1



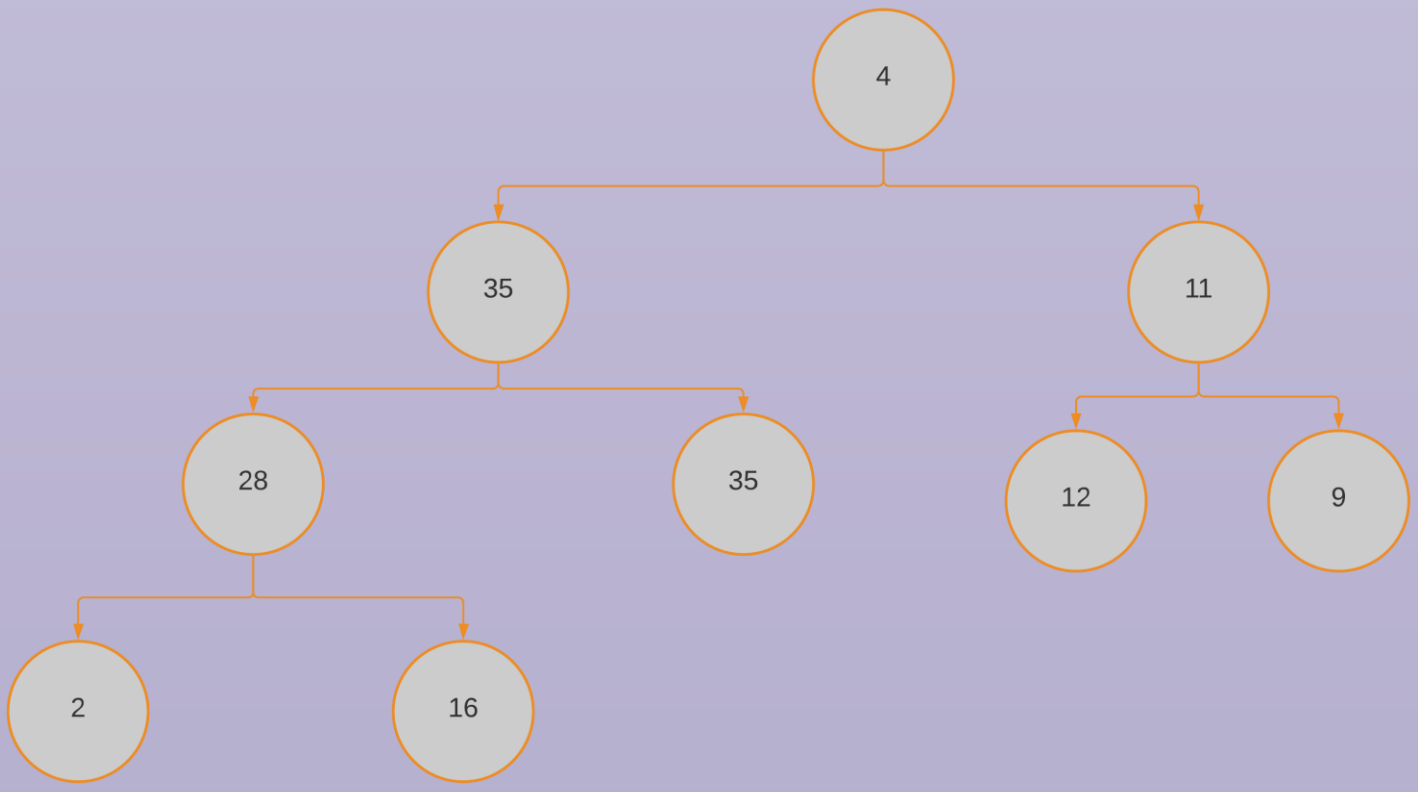
.2



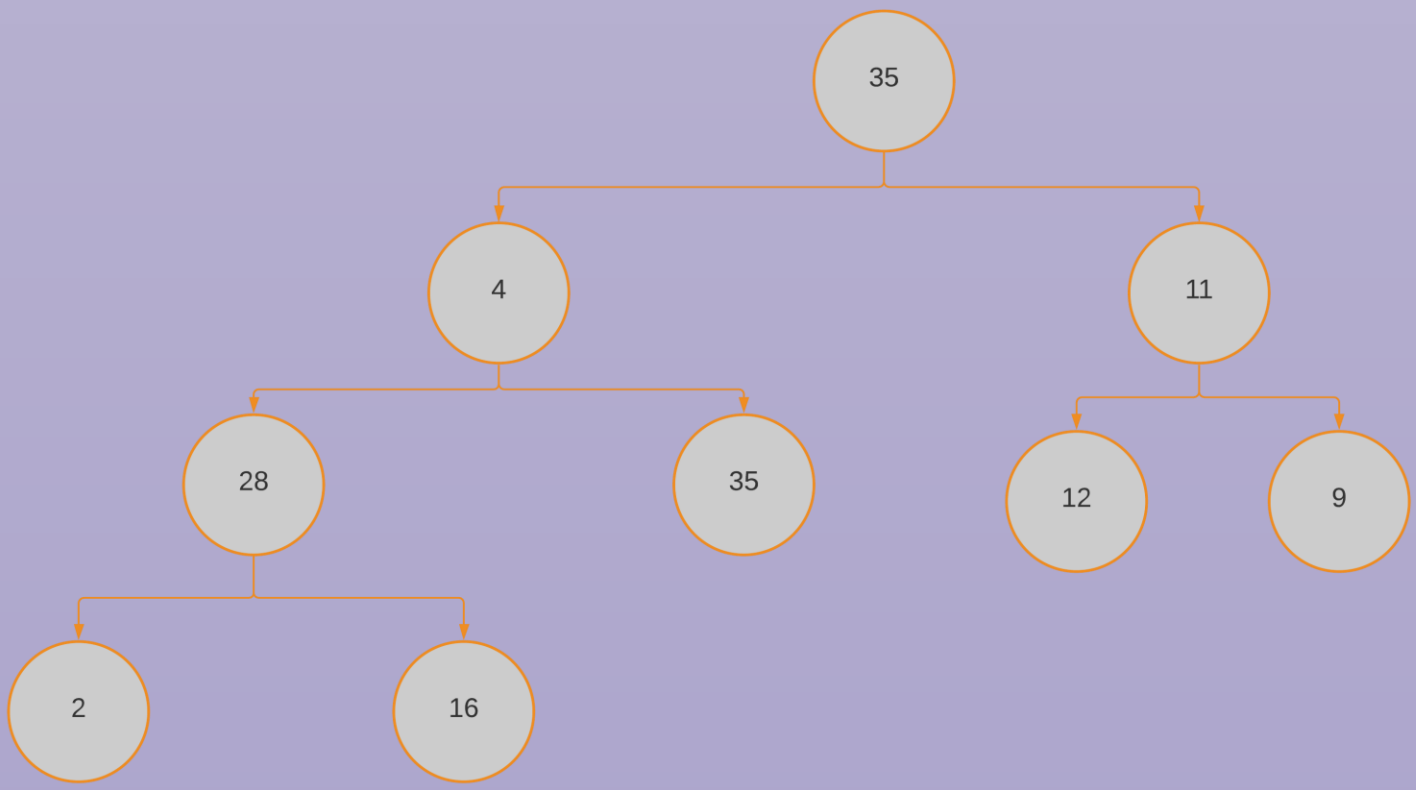
تحلیل و طراحی الگوریتم ها

تمرین سری اول

.3



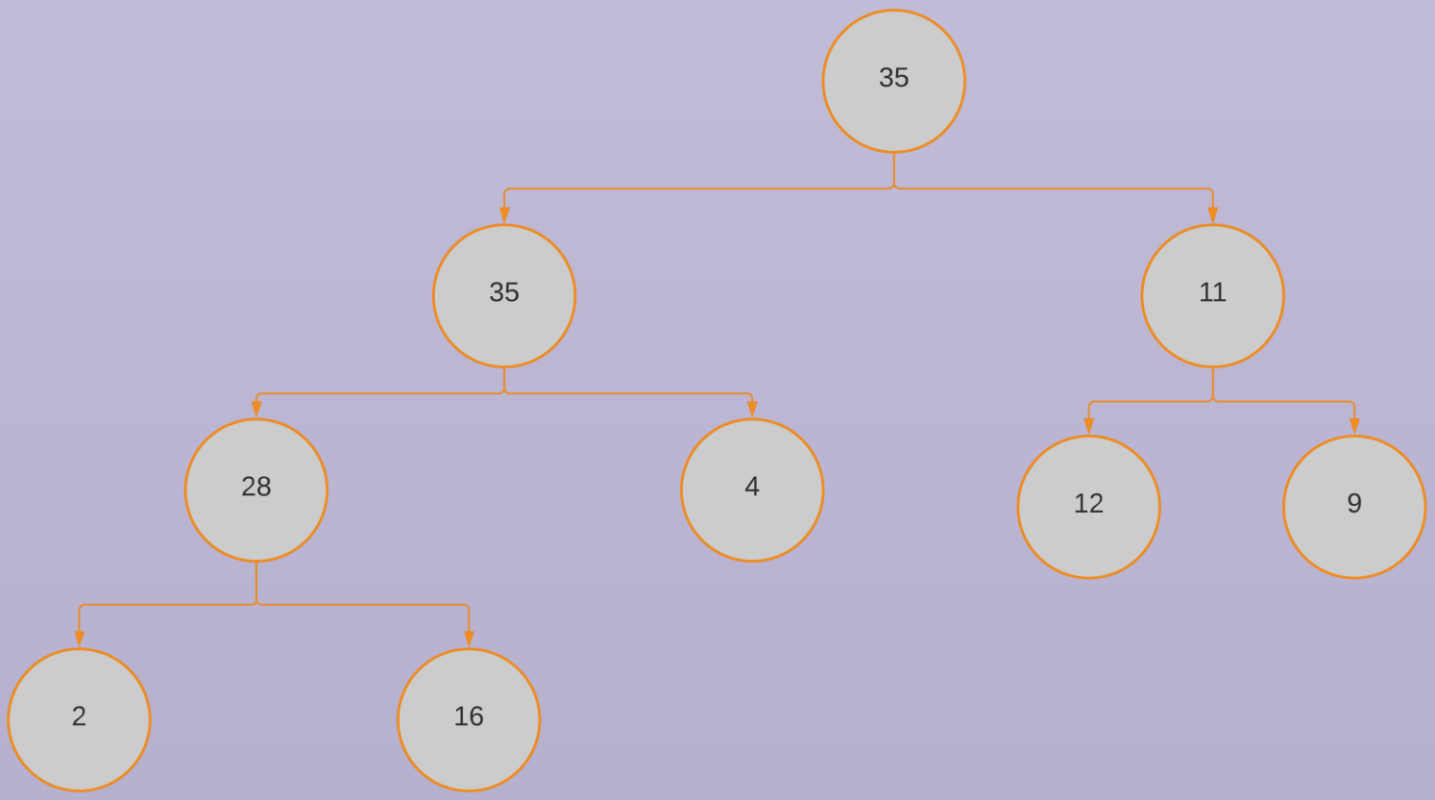
.4



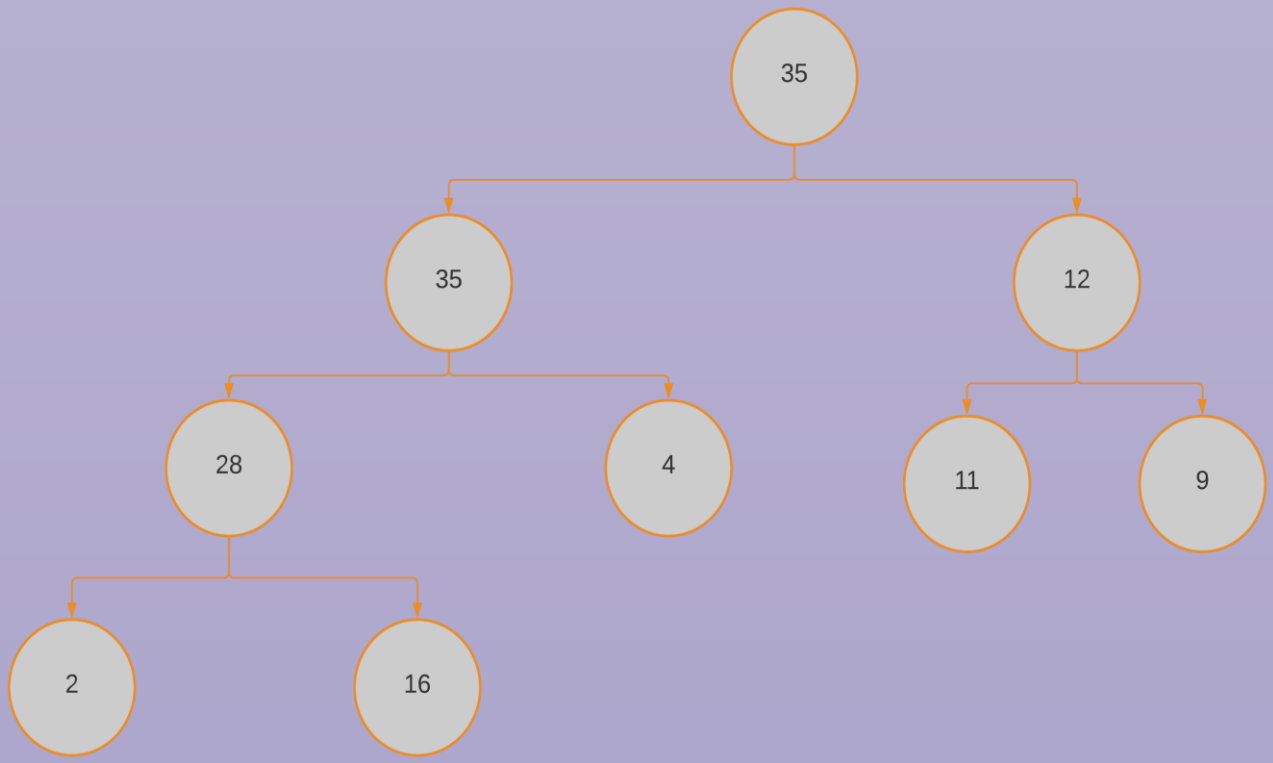
تحلیل و طراحی الگوریتم ها

تمرین سری اول

.5



.6



تحلیل و طراحی الگوریتم ها
تمرین سری اول

C. What is the running time of Heap Sort?

n log n که حد محکم آن بدین صورت است :

$$\sum_{h=0}^{\lfloor \lg n \rfloor} o(h)\, n_h$$

برای مرتب‌کردن عناصر با استفاده از درخت heap، نیاز به n عمل حذف گره ریشه از درخت (یا عمل pop) وجود دارد. عمل حذف گره ریشه در درخت heap خود از مرتبه‌ی Θ(log n) است. در نتیجه کل این عملیات از مرتبه‌ی Θ(n log n) خواهد بود.

D. What is the worst case running time of Heap Sort ? (Why?)

رتبه زمانی این الگوریتم nlog n است که برای ساخت هیپ n و برای هر یک از n-1 نود، فراخوانی heapify می شود log n که ترکیب این دو در بدترین حالت(O) n log n می شود.

E. Implement Heap Sort using java programming language and send It to quera. (send your file please)

Ok Do I do that !!!

تحلیل و طراحی الگوریتم ها
تمرین سری اول

F. What are the advantages and disadvantages of Heap Sort?

مزایا	معایب
استفاده از حداقل حافظه	مرتب سازی سریع دربسیاری موارد پرکابرد تر از این مرتب سازی است
می تواند به صورت در جا نیز استفاده شود	نیازمنده به یک درختی از عناصر است
کمک برای یافتن بیشترین و کمترین عنصر	معمولا زمان بیشتری برای مرتب سازی نسبت به بقیه الگوریتم ها نیاز دارد

G. Explain the memory usage of Heap Sort. (at most on 5 lines)

این الگوریتم بسته به طریقه طراحی آن می تواند به صورت الگوریتم درجا نیز عمل کند، این بدان معناست که میزان استفاده از حافظه آن کم است زیرا جدا از آنچه برای نگهداری لیست اولیه موارد مرتب سازی لازم است ، برای کار به فضای حافظه اضافی نیاز نیست .