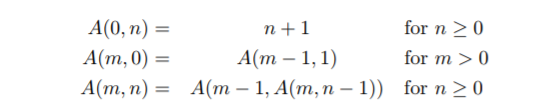
به نام خدا

|  |  |  |  |
| --- | --- | --- | --- |
| مدرس: مظفر بگ محمدي | دانشگاه ايلام | ترم اول سال تحصيلي 04-03 | سری اول تمرینات ساختمان داده |

1. یک برنامه ی بازگشتی بنویسید که تابع آکرمان را محاسبه کند. تعریف این تابع به صورت زیر است:



1. یک برنامه بازگشتی بنویسید یک آرایه از اعداد طبیعی دریافت کند و به صورت زیر عمل کند :

Input: A = {1, 2, 3, 4, 5}

Output: [48]

[20, 28]

[8, 12, 16]

[3, 5, 7, 9]

[1, 2, 3, 4, 5]

راهنمایی: در اولین چرخش تمام عناصر هستند و سپس هر عنصر با عنصر بعدی خود جمع میشود

Guidance :

[48]

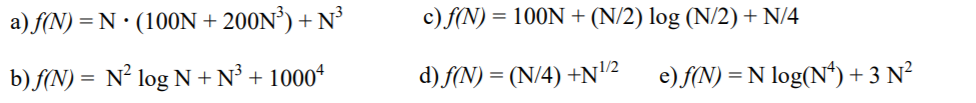
[20, 28] 🡪(20 + 28 = 48)

[8, 12, 16] 🡪 (8 + 12 = 20, 12 + 16 = 28)

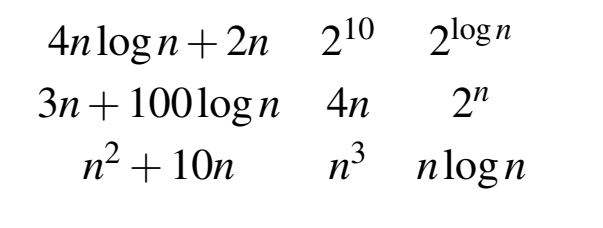
[3, 5, 7, 9] 🡪 (3 + 5 = 8, 5 + 7 = 12, 7 + 9 = 16)

[1, 2, 3, 4, 5] 🡪 (1 + 2 = 3, 2 + 3 = 5, 3 + 4 = 7, 4 + 5 = 9)

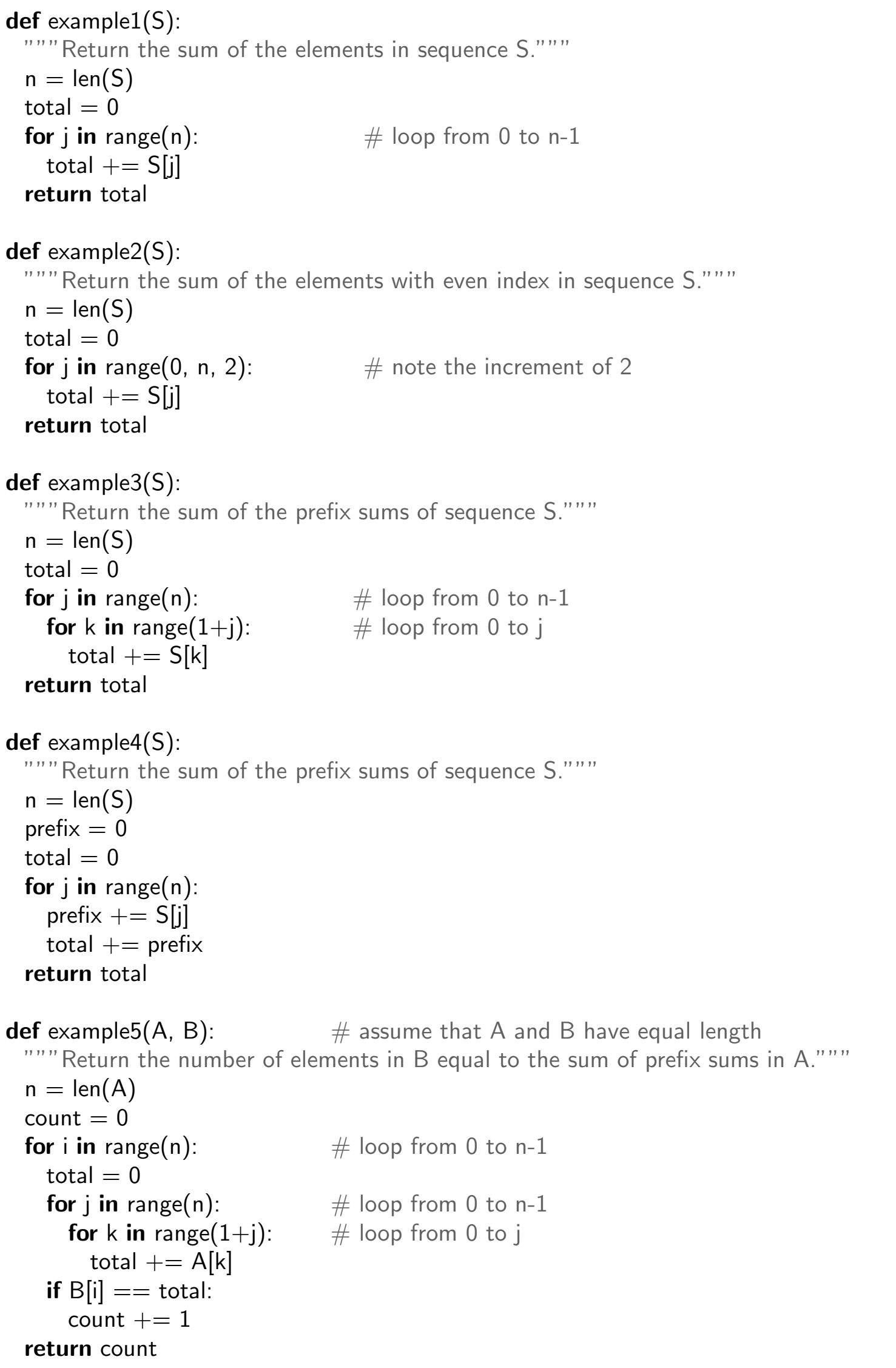
1. نشان دهید اگر f1(n)=Ω(g1(n)) و f2(n)=Ω(g2(n)) باشد، آنگاه f1(n)×f2(n)= Ω(g1(n)×g2(n)) است.
2. برای هر یک از عبارات زیر کمترین O ممکن را بدست آورید:

****

1. نشان دهید برای یک تابع چند جمله ای مثل p(n)، پیچیدگی محاسباتی تابع log p(n) از درجه ی O(log n) است.
2. توابع زیر را از نظر پیچیدگی محاسباتی مرتب کنید:



1. با استفاده از تئوری Master پیچیدگی محاسباتی (زمانی) حالات زیر را تعیین کنید:
   1. **T(n) = 3T(n/3) + n3**
   2. **T(n) = 3T(n/2) + n**
   3. **T(n) = 2T(n/3) +3n2+n**
   4. **T(n) = T() + 1**
   5. **T(n) = 8T() +**
   6. **T(n) = T() +**
2. پیچیدگی محاسباتی قطعه کدهای زیر را محاسبه کنید:



1. یک برنامه­ی بازگشتی بنویسید که بتواند بزرگترین عنصر یک آرایه را پیدا کند. سپس با استفاده از تئوری master پیچیدگی زمانی برنامه را حساب کنید.
2. فرض کنيد که الگوريتمهاي T و C براي حل يک مساله مشابه با هم رقابت مي کنند. پيچيدگي زماني T از رابطه T(n)=7T(n/2)+n2 پيروي مي کند و پيچيدگي زماني C از رابطه T(n)=aT(n/4)+n2 پيروي مي کند. بزرگترين مقدار a را که به ازاي آن C سريعتر است را بدست آوريد.
3. نشان دهید که از درجه O(nlogn) است.
4. پیچیدگی محاسباتی کد زیر را محاسبه کنید:

