HW-01

Algorithm Design - Spring 2026

Instructor: Dr. Amin Eskandari

Head Teaching Assistants: Hamid Namjoo, Amir Hossein Hemati

Teaching Assistants: Ali Nikvan, Viyana Hasanbeigi, Nima Soltani, Reza Rezaie, Maryam Meidanshahi , Babak Janfaza , Fatemeh Dehgan , Elham Mosavi

---

سوال ۱ :

💰📚عمارت وین و چالش سکه‌های آلفرد پنی‌ورث

در سال‌های دور، در باشکوه‌ترین شب عمارت وین، جشنی بزرگ برگزار شد. پس از پایان مراسم، Alfred Pennyworth که مسئول تدارکات و پرداخت‌ها بود، باید انعام کارکنان را پرداخت کند.

او برای پرداخت این انعام به مبلغ nتومان نیاز دارد.
اما یک مشکل وجود دارد...

در آن زمان، سیستم بانکی تنها سکه‌هایی با مقادیر زیر در اختیار داشت:

۱ ، ۵ ، ۱۰ ، ۲۰ ، ۱۰۰ تومان

آلفرد که فردی دقیق و منظم است، می‌خواهد مبلغ موردنظر را با کمترین تعداد سکه ممکن پرداخت کند تا کار سریع‌تر انجام شود.

🎯وظیفه شما

برنامه‌ای طراحی کنید که با دریافت عدد صحیح مثبت n،
حداقل تعداد سکه‌های لازم برای ساخت مبلغ nتومان را محاسبه کند.

📥ورودی

یک عدد صحیح مثبت nکه نشان‌دهنده مبلغ موردنظر (به تومان) است.

📤خروجی

یک عدد صحیح که نشان‌دهنده حداقل تعداد سکه‌های لازم برای پرداخت مبلغ n تومان است.

📌قوانین مسئله

- **مقدار nیک عدد صحیح مثبت است.**
- **فقط از سکه‌های با مقادیر ۱، ۵، ۱۰، ۲۰، ۱۰۰ تومان می‌توان استفاده کرد.**
- **هدف، کمینه کردن تعداد کل سکه‌ها است.**
- **هیچ محدودیتی در تعداد هر نوع سکه وجود ندارد.**

نمونه ورودی و خروجی

ورودی:

۱۲۵

خروجی:

۳

سوال ۲ :

🗜️📚سامانه فشرده‌سازی و چالش رمزگشایی هافمن

در یک مرکز پیشرفته‌ی پردازش داده، پیام‌های متنی با استفاده از الگوریتم
David A. Huffman
فشرده‌سازی شده‌اند.

سیستم از **کد هافمن (Huffman Coding)** برای کاهش حجم پیام‌ها استفاده می‌کند. اکنون نسخه‌ی فشرده‌شده‌ی یک پیام در اختیار شماست، اما برای خواندن آن باید ابتدا آن را رمزگشایی کنید.

🎯مأموریت شما

برنامه‌ای طراحی کنید که:

۱. با استفاده از فراوانی کاراکترها، **درخت هافمن** را بسازد.
۲. با استفاده از درخت ساخته‌شده، رشته‌ی دودویی داده‌شده را **رمزگشایی** کند.
۳. در نهایت، **پیام اصلی** را چاپ کند.

🌳قوانین ساخت درخت هافمن

در هنگام ساخت درخت باید دقیقاً قوانین زیر رعایت شود:

۱. در هر مرحله، **دو نودی که کمترین فراوانی را دارند** انتخاب می‌شوند.
۲. اگر فراوانی دو نود برابر باشد، نودی که کاراکتر آن از نظر **ASCIIکوچک‌تر** است در اولویت قرار می‌گیرد.
۳. نود با اولویت کمتر، **فرزند چپ (کد ۰)** و نود دیگر **فرزند راست (کد ۱)** می‌شود.
۴. نودهای داخلی فاقد کاراکتر هستند و با نماد #در نظر گرفته می‌شوند.

📥 ورودی

ورودی به‌ترتیب شامل موارد زیر است:

N
C1 F1
C2 F2
...
CN FN
Encoded_string

توضیح پارامترها:

- **N** : تعداد کاراکترها
- **Ci** : یک کاراکتر انگلیسی کوچک
- **Fi** : فراوانی آن کاراکتر
- **Encoded_string** : رشته‌ی دودویی رمزگذاری‌شده

📤 خروجی

Decoded_string

رشته‌ی رمزگشایی‌شده (پیام اصلی).

📌 قوانین کلی

- تمام کاراکترها انگلیسی کوچک هستند.
- فراوانی‌ها اعداد صحیح مثبت هستند.
- ساخت درخت باید دقیقاً مطابق قوانین اولویت ذکرشده انجام شود.
- فرض کنید ورودی معتبر است.

سوال ۳ :

🛍️💰چالش خرید هوشمند علی

علی قصد دارد با مقدار محدودی پول، چند وسیله بخرد.
او اطلاعات کاملی از ارزش وسایل، ظرفیت کیف پولش و هدف موردنظرش دارد؛
اما تصمیم نهایی او به یک بررسی الگوریتمی دقیق بستگی دارد!

🎯مأموریت شما

برنامه‌ای طراحی کنید که براساس داده‌های ورودی، تصمیم بگیرد علی چه کاری انجام دهد.

📦اطلاعات در اختیار علی

- **Nعدد که هرکدام نشان‌دهنده‌ی ارزش یک وسیله هستند.**
- **یک عدد Wکه نشان‌دهنده‌ی ظرفیت کیف پول علی (حداکثر پولی که می‌تواند خرج کند) است.**
- **یک عدد Tکه هدف علی برای رسیدن به مجموع ارزش خاصی است.**
- **kعدد که مقادیر اسکناس‌های موجود را نشان می‌دهند.**

🧠مرحله اول: بررسی سه‌مجموع(Three-Sum)

ابتدا باید بررسی شود:
آیا سه وسیله متفاوت وجود دارند که مجموع ارزششان دقیقاً برابر Tباشد؟

✅اگر چنین سه‌تایی وجود داشت:

علی وارد مرحله دوم می‌شود:

🎒کوله‌پشتی کسری(Fractional Knapsack)

- **علی می‌تواند بخشی از هر وسیله را بخرد.**

- **ارزش هر وسیله همان عدد داده‌شده است.**
- **وزن هر وسیله برابر است با عدد فیبوناچی متناظر با اندیس آن وسیله (اندیس‌ها از ۱ شروع می‌شوند).**
- **ظرفیت کوله‌پشتی برابر Wاست.**
- **باید بیشترین ارزش قابل حمل محاسبه شود.**

❌اگر چنین سه‌تایی وجود نداشت:

علی باید مبلغ Wرا با استفاده از کمترین تعداد اسکناس از مقادیر داده‌شده خرد کند

مسئله‌ی خرد کردن پول.(Coin Change Minimum Coins)

اگر امکان ساخت دقیق مبلغ W وجود نداشت، خروجی برابر 1-خواهد بود.

📥ورودی

n

a1 a2 ... an

W

T

k

c1 c2 ... ck

📌توضیح ورودی‌ها

- **n : تعداد وسایل**
- **ai : ارزش هر وسیله**
- **W : ظرفیت کیف پول**
- **T : مقدار هدف برای مسئله سه‌مجموع**
- **k : تعداد اسکناس‌ها**
- **ci : مقادیر اسکناس‌ها**

📤خروجی

اگر سه عددی وجود داشت که مجموعشان T باشد:

YES

<maximum_value>

- **در خط اولYES :**
- **در خط دوم: بیشترین ارزش قابل حمل در کوله‌پشتی کسری (با دقت دو رقم اعشار)**

در غیر این صورت:

NO

<minimum_number_of_bills>

- **در خط اولNO :**
- **در خط دوم: حداقل تعداد اسکناس برای خرد کردنW**
- **اگر ممکن نبود، 1-چاپ شود.**

📌قوانین کلی

- **اندیس وسایل از 1شروع می‌شود (برای محاسبه عدد فیبوناچی).**
- **اعداد ورودی همگی صحیح و مثبت هستند.**
- **هر وسیله فقط یک‌بار در بررسی سه‌مجموع می‌تواند استفاده شود (باید سه وسیله متفاوت باشند).**
- **در کوله‌پشتی کسری می‌توان بخشی از وسیله را برداشت.**

سوال ۴ :

🧩⚙️سامانه پردازشی هوشمند و چالش انتخاب کارها

در یک مرکز پردازش پیشرفته، سیستمی وجود دارد که باید از میان nکار مختلف، مجموعه‌ای بهینه را انتخاب کند.
هر کار دارای مشخصات دقیق زمانی، مالی و یک رشته‌ی رمزگذاری‌شده است.
سیستم دارای بودجه‌ای محدود به مقدار Bاست و باید به شکلی تصمیم بگیرد که بیشترین سود ممکن حاصل شود — آن هم با رعایت تمام محدودیت‌ها.

📥مشخصات هر کار

برای هر کار اطلاعات زیر داده می‌شود:

- **si : زمان شروع مجاز**
- **fi : زمان پایان مجاز**
- **wi : سود انجام کامل کار**
- **ci : هزینه انجام کامل کار**
- **di : رشته رمزگذاری‌شده (Encoded String)**

قوانین مسئله

زمان‌بندی(Scheduling)

- **اگر کاری انتخاب شود، باید کاملاً در بازه‌ی [si , fi]اجرا شود.**
- **هیچ دو کاری نباید هم‌پوشانی زمانی داشته باشند.**

دیکدینگ(Decoding)

رشته‌ی diبه صورت زیر رمزگشایی می‌شود:

- **هر عدد kبه معنی تکرار کاراکتر قبلی به اندازه‌ی kاست.**

مثال:

a3b2 → aaabb

کوله‌پشتی کسری(Fractional Knapsack)

- **می‌توان بخشی از یک کار را انجام داد.**
- **اگر فقط xدرصد از کار انجام شود:**

سود = x × wi

هزینه = x × ci

- **طول رشته‌ی دیکد شده نیز به همان نسبت در نظر گرفته می‌شود.**

محدودیت خرد کردن پول(Coin Change Constraint)

- **هزینه‌ی کل انتخاب‌شده باید بتواند دقیقاً با سکه‌های زیر پرداخت شود:**

۱ ، ۳ ، ٤

- **اگر پرداخت دقیق ممکن نباشد، آن انتخاب نامعتبر است.**

📥ورودی

n B

s1 f1 w1 c1 d1

s2 f2 w2 c2 d2

...

sn fn wn cn dn

توضیح ورودی‌ها:

- **n : تعداد کارها**
- **B : بودجه کل سیستم**
- **برای هر کار:**
  - **si, fi : بازه زمانی مجاز**
  - **wi : سود کامل**
  - **ci : هزینه کامل**

- **di : رشته رمزگذاری‌شده**

📤خروجی

Maximum_Profit

بیشترین سود قابل دستیابی با رعایت تمام قوانین بالا.

📌قوانین کلی

- **کارها نباید هم‌پوشانی زمانی داشته باشند.**
- **مجموع هزینه نباید از Bبیشتر شود.**
- **هزینه نهایی باید دقیقاً با سکه‌های ۱، ۳، ٤ قابل پرداخت باشد.**
- **انجام کار می‌تواند به صورت کامل یا کسری باشد.**
- **تمامی مقادیر ورودی اعداد صحیح مثبت هستند.**