۱:.Arrayو List چه تفاوتی دارند؟

تفاوت Array و List به طور خلاصه:

\*Array (آرایه): طول ثابت، نوع داده‌ای یکسان، دسترسی سریع به عناصر.

\*List (لیست): طول متغیر، می‌تواند انواع داده‌ای مختلف را در خود جای دهد، انعطاف‌پذیری بیشتر.

۲:.Dictionaryدر Python چگونه کار میکند؟

در پایتون: \*\*Dictionary(دایره‌المعارف) یک نوع ساختار داده‌ای است که به صورت زوج‌های کلید-مقدار عمل می‌کند. این ویژگی‌ها نحوه کارکرد آنها را به طور خلاصه توضیح می‌دهد:

1. \*\*ساختار داده‌ای: مجموعه‌ای از کلیدها و مقادیر است که هر کلید باید یکتا باشد.

2. \*\*دسترسی سریع: می‌توانید به مقادیر با استفاده از کلیدها به سرعت دسترسی پیدا کنید.

3. \*\*متنوع بودن:مقادیر می‌توانند از انواع داده‌ای مختلف (مثل عدد، رشته، لیست و غیره) باشند.

4. \*\*قابل تغییر:\*\* می‌توانید به راحتی مقادیر را اضافه، حذف یا تغییر دهید.

مثال:

```python

My\_dict = {'name': 'Alice', 'age': 25}

Print(my\_dict['name']) # خروجی: Alice

```

به طور کلی، دیکشنری‌ها برای ذخیره و مدیریت داده‌ها به صورت سازمان‌یافته و قابل دسترسی سریع بسیار مفید هستند.

۳:.Tupleو List چه تفاوتی دارند؟

\*\*Tuple و \*\*List در پایتون هر دو ساختارهای داده‌ای برای ذخیره‌سازی مجموعه‌ای از عناصر هستند، اما تفاوت‌های کلیدی زیر را دارند:

1. \*\*نوع داده:

- \*\*Tuple (تاپل): غیرقابل تغییر (immutable) است؛ یعنی پس از ایجاد، عناصر آن قابل تغییر نیستند.

- \*\*List (لیست): قابل تغییر (mutable) است؛ می‌توانید عناصر آن را تغییر دهید یا به آن‌ها اضافه یا از آن‌ها حذف کنید.

2. \*\*نحو:

- \*\*Tuple:با پرانتز گرد تعریف می‌شود.

- \*\*List: با پرانتز مربع تعریف می‌شود

3. \*\*عملکرد:

- \*\*Tuple:معمولاً برای ذخیره‌سازی داده‌هایی که نباید تغییر کنند، استفاده می‌شود و به دلیل غیرقابل تغییر بودن، می‌تواند عملکرد بهتری در برخی موارد داشته باشد.

- \*\*List: مناسب برای ذخیره‌سازی داده‌هایی است که باید تغییر کنند.

\*\*مثال:

```python

My\_list = [1, 2, 3] # لیست

My\_tuple = (1, 2, 3) # تاپل

```

به طور خلاصه، \*\*Listبرای داده‌های تغییرپذیر و \*\*Tuple برای داده‌های غیرقابل تغییر استفاده می‌شود.

۴:در Python چرا برای حذف دادههای تکراری استفاده می شود

در پایتون، `Set` (مجموعه) برای حذف داده‌های تکراری استفاده می‌شود زیرا:

1. عناصر یکتا: `Set` فقط می‌تواند عناصر یکتا (منحصربه‌فرد) را در خود نگه دارد. اگر سعی کنید یک عنصر تکراری به آن اضافه کنید، `Set` آن را نادیده می‌گیرد.
2. سرعت:استفاده از `Set` برای حذف تکراری‌ها بسیار سریع‌تر از روش‌های دیگر (مانند استفاده از حلقه و بررسی تکراری‌ها) است، به خصوص برای داده‌های بزرگ.

به طور خلاصه، `Set` با خاصیت داشتن عناصر یکتا و سرعت بالا، ابزاری کارآمد برای حذف داده‌های تکراری در پایتون است.

۵:.Stackو Queue چه تفاوتی دارند؟

\*\*Stack (پشته) و \*\*Queue (صف) هر دو ساختارهای داده‌ای خطی هستند که برای ذخیره و بازیابی عناصر به ترتیب خاصی استفاده می‌شوند، اما تفاوت اصلی آن‌ها در نحوه دسترسی به عناصر است:

* \*\*Stack (پشته): از اصل\*LIFo (Last-In, First-Out) یا "آخرین ورودی، اولین خروجی" پیروی می‌کند. به این معنی که آخرین عنصری که به پشته اضافه شده، اولین عنصری است که از آن خارج می‌شود. تصور کنید یک دسته بشقاب روی هم چیده‌اید؛ آخرین بشقابی که روی دسته گذاشته‌اید، اولین بشقابی است که برمی‌دارید.
* \*Queue (صف): از اصل \*FIFO\* (First-In, First-Out) یا "اولین ورودی، اولین خروجی" پیروی می‌کند. به این معنی که اولین عنصری که به صف اضافه شده، اولین عنصری است که از آن خارج می‌شود. تصور کنید در یک صف نانوایی ایستاده‌اید؛ اولین نفری که در صف قرار گرفته، اولین نفری است که نان دریافت می‌کند.

به طور خلاصه:

\*\*Stack:\* LIFO (آخرین ورودی، اولین خروجی)

\*\*Queue:\*\* FIFO (اولین ورودی، اولین خروجی)

۶:.Table Hashچیست و چرا کاربرد دارد؟

\*\*Hash Table (جدول درهم‌سازی)\* یک ساختار داده‌ای است که برای ذخیره و بازیابی سریع داده‌ها استفاده می‌شود.

نحوه کار:

1. \*درهم‌سازی (Hashing):\* یک تابع درهم‌سازی (Hash Function) کلید (Key) هر داده را به یک اندیس (Index) در جدول تبدیل می‌کند.

2. ذخیره‌سازی:داده در اندیس محاسبه‌شده در جدول ذخیره می‌شود.

3. \*بازیابی:\* برای بازیابی داده، دوباره با استفاده از تابع درهم‌سازی، اندیس آن محاسبه و داده مورد نظر بازیابی می‌شود.

\*چرا کاربرد دارد؟\*

\* \*سرعت بالا:\* بازیابی داده‌ها در جدول درهم‌سازی به طور متوسط بسیار سریع‌تر از روش‌های دیگر (مانند جستجو در آرایه یا لیست) است. این سرعت بالا به دلیل استفاده از تابع درهم‌سازی برای دسترسی مستقیم به محل ذخیره‌سازی داده است

\* \*پیاده‌سازی آسان:\* پیاده‌سازی جدول درهم‌سازی نسبتاً ساده است.

\* کارایی بالا:برای مجموعه‌های بزرگ داده، جدول درهم‌سازی کارایی بسیار خوبی دارد.

به طور خلاصه، Hash Table یک ساختار داده‌ای سریع و کارآمد برای ذخیره و بازیابی داده‌ها بر اساس کلید است.

۷: تفاوتی چه B-Tree وBinary Tree دارند؟

\*\*Binary Tree (درخت دودویی) و \*B-Tree\*\* هر دو ساختارهای داده‌ای درختی هستند، اما تفاوت‌های اساسی دارند:

\*\*Binary Tree (درخت دودویی):

\* هر گره حداکثر دو فرزند دارد (فرزند چپ و فرزند راست).

\* برای جستجو، درج و حذف داده‌ها استفاده می‌شود.

\* متعادل نبودن درخت می‌تواند باعث کاهش کارایی شود (در بدترین حالت، زمان جستجو به O(n) می‌رسد).

\*\*B-Tree:

\* هر گره می‌تواند تعداد زیادی فرزند داشته باشد (به این تعداد "درجه" درخت گفته می‌شود).

\* به‌طور ویژه برای ذخیره‌سازی داده‌ها روی دیسک (مانند پایگاه داده‌ها) طراحی شده است.

\* همیشه متعادل است و عمق آن کم است، بنابراین زمان جستجو، درج و حذف داده‌ها بسیار سریع است.

\*\*تفاوت‌های کلیدی (به صورت خلاصه):

| ویژگی | Binary Tree | B-Tree |

| ------------- | -------------------------- | ------------------------ |

| حداکثر فرزندان | 2 | تعداد زیاد (درجه درخت) |

| کاربرد | پیاده‌سازی‌های عمومی درخت | ذخیره‌سازی داده روی دیسک |

| تعادل | ممکن است متعادل نباشد | همیشه متعادل است |

| عمق | می‌تواند زیاد باشد | کم |

به طور خلاصه، B-Tree یک نوع خاص از درخت است که برای کارایی بالا در ذخیره‌سازی و بازیابی داده‌ها روی دیسک طراحی شده است، در حالی که Binary Tree یک ساختار داده‌ای درختی عمومی‌تر است.

۸:چرا Structure Data Graph برای شبکههای اجتماعی استفاده می شود

ساختار داده‌ای \*Graph (گراف) برای شبکه‌های اجتماعی بسیار مناسب است، زیرا به طور طبیعی ارتباطات و روابط بین افراد را مدل‌سازی می‌کند.

دلایل استفاده از Graph در شبکه‌های اجتماعی:

1. مدل‌سازی روابط: در یک شبکه اجتماعی، افراد (یا "گره‌ها") با یکدیگر رابطه دارند (مانند دوستی، دنبال کردن، همکار بودن و غیره). گراف به سادگی این روابط را با استفاده از "یال‌ها" بین گره‌ها نشان می‌دهد.

2. تحلیل شبکه‌های اجتماعی: با استفاده از الگوریتم‌های گراف، می‌توان اطلاعات مفیدی از ساختار شبکه استخراج کرد. به عنوان مثال:

یافتن افراد تاثیرگذار (گره‌های با بیشترین ارتباط).

تشخیص گروه‌های دوستان (اجتماعات در گراف).

پیشنهاد دوستان جدید (بر اساس ارتباطات مشترک).

1. جستجوی کارآمد: گراف امکان جستجوی سریع روابط بین افراد را فراهم می‌کند.

به طور خلاصه:

گراف به خوبی روابط بین افراد را مدل‌سازی می‌کند.

الگوریتم‌های گراف امکان تحلیل شبکه‌های اجتماعی و استخراج اطلاعات مفید را فراهم می‌کنند.

جستجو در گراف برای یافتن روابط کارآمد است.

I .Programming Dynamic:9چرا در حل مسائل پیچیده کاربرد دارد؟

برنامه‌نویسی پویا (Dynamic Programming) در حل مسائل پیچیده کاربرد دارد، چون:

1. شکستن مسئله به زیرمسئله‌های کوچک‌تر: مسائل پیچیده را به زیرمسئله‌های کوچک و قابل حل تبدیل می‌کند.
2. ذخیره‌سازی نتایج: نتایج زیرمسئله‌ها را ذخیره می‌کند تا در صورت نیاز مجدد، محاسبه تکراری انجام نشود (Memoization).
3. بهینه‌سازی:با پیدا کردن بهترین راه حل از بین راه حل‌های مختلف، به بهینه‌ترین جواب می‌رسد.

به طور خلاصه:

تقسیم مسئله به بخش‌های کوچک‌تر

جلوگیری از محاسبات تکراری با ذخیره‌سازی نتایج

یافتن بهترین راه حل از بین گزینه‌های موجود

این سه ویژگی باعث می‌شوند برنامه‌نویسی پویا برای حل مسائل پیچیده با زیرمسئله‌های همپوشان بسیار کارآمد باشد.

۱۰:.Recursionچیست و چرا در الگوریتمهای پیشرفته استفاده می شود

\*\*Recursion (بازگشت) یک تکنیک برنامه‌نویسی است که در آن یک تابع، خودش را صدا می‌زند تا یک مسئله را حل کند.

\*\*چرا در الگوریتم‌های پیشرفته استفاده می‌شود؟

1. \*\*حل مسائل قابل تقسیم: برای مسائلی که می‌توانند به زیرمسئله‌های کوچکتر و مشابه تقسیم شوند، راه حل بازگشتی معمولاً ساده‌تر و خواناتر است.
2. \*\*ساختارهای داده‌ای بازگشتی: برای کار با ساختارهای داده‌ای مانند درخت‌ها و گراف‌ها که ذاتاً بازگشتی هستند، بسیار مناسب است.
3. \*\*الگوریتم‌های تقسیم و غلبه (Divide and Conquer):در الگوریتم‌هایی مانند مرتب‌سازی سریع (Quicksort) و ادغام (Mergesort) که مسئله را به بخش‌های کوچکتر تقسیم و سپس آن‌ها را حل می‌کنند، بازگشت نقش کلیدی دارد.

\*\*به طور خلاصه:

تابع خودش را صدا می‌زند.

برای مسائل قابل تقسیم و ساختارهای داده‌ای بازگشتی مناسب است.

در الگوریتم‌های تقسیم و غلبه استفاده می‌شود.

با این حال، باید توجه داشت که استفاده نادرست از بازگشت می‌تواند منجر به مشکلات کارایی (مانند سرریز پشته) شود، بنابراین باید با دقت مورد استفاده قرار گیرد.