Subject:

درس : مباحث ویژه

موضوع : Data Structures and Algorithms

نام دانشجو :

نرجس سکل پور

کبری رحمی زاده

نام استاد : محمد احمدزاده

Subject: (ایضس) Date:

A. Array و List چه تفاوتی دارند؟

- تفاوت اصلی بین Array و List در زبان‌های برنامه‌نویسی مثل جاوا و پایتون به این صورت هست:

1. اندازه‌گیری:

Array

- اندازه‌اش ثابت هست و وقتی که یک آرایه را تعریف می‌کنید باید اندازه‌اش را مشخص کنید و نمی‌توانید بعداً تغییرش بدهید.

- List: اندازه‌اش متغیر هست و می‌توانید به راحتی عناصر جدید را اضافه کنید یا حذف کنید.

2. نوع داده:

Array

- معمولاً نوع داده‌اش یکنواخت هست، یعنی همه عناصر باید از یک نوع باشند.

- List: می‌تونه شامل انواع مختلف از داده‌ها باشد (بسته به زبان برنامه‌نویسی).

3. عملکرد:

Array

- دسترسی به عناصرش سریع‌تر هست چون به صورت مستقیم به آدرس حافظه دسترسی دارد.

- List: ممکن کمی کندتر باشد چون ممکنه نیاز به جابجایی یا تغییر اندازه داشته باشد.

Subject: Date:

B- Dictionary در python چیست و کاربرد آن چیست؟

در پایتون، دیکشنری (Dictionary) یک نوع داده‌ای هست که به ما اجازه می‌دهد اطلاعات را به صورت جفت‌های کلید-ارزش ذخیره کنیم. یعنی هر کلید یک مقدار خاص را به خودش نسبت می‌دهد.

- ساختار دیکشنری به این شکل هست:

```
my_dict = {
    "کلید1" : "مقدار1"
    "کلید2" : "مقدار2"
    "کلید3" : "مقدار3"
}
```

- برای دسترسی به مقدارها می‌توان از کلیدها استفاده کنی:

```
print(my_dict["کلید1"])  # خروجی: مقدار1
```

می‌توان مقدارها را هم تغییر بدی یا اضافه کنید؟

```
my_dict["کلید4"] = "مقدار4"
print(my_dict)
```

- برای حذف یک کلید و مقدارش می‌توان از del استفاده کنید:

```
del my_dict["کلید2"]
print(my_dict)
```

Subject: Date:

C- List و Tuple چه تفاوتی دارند؟

تفاوت های اصلی بین List و Tuple در پایتون به این صورت هستند:

1. تغییر پذیری: List تغییر پذیر (mutable) هستن یعنی می تونن عناصرشون رو تغییر بدی اضافه کنی یا حذف کنی.

- Tuple: یعنی بعد از ایجاد، (immutable) غیر تغییر پذیر ن یعنی کل عناصرش رو تغییر بدی هستن

2. نحوه تعریف: List - با استفاده از براکت ها [] تعریف می شوند. مثلاً my_list = [1,2,3]

- Tuple - با استفاده از پرانتز ها () تعریف می شوند. مثلاً: my_tuple = (1,2,3)

3. کاربرد: - List - معمولاً برای مجموعه های قابل تغییر استفاده می شوند.

- Tuple - بیشتر برای مجموعه های ثابت و داده های غیر قابل تغییر به کار میرند.

4. عملکرد: Tuple به دلیل غیر تغییر پذیری، معمولاً سریع تر از لیست ها عمل می کنن.

Subject: Date:

D. در Python از set برای حذف داده های تکراری استفاده می شود.

- در set یک نوع داده ای هست که می توان به راحتی از داده ای تکراری خلاص شد.

به عبارت دیگر، وقتی شما یک لیست یا هر نوع داده ای دیگری را به set تبدیل می کنی تمام عناصر تکراری حذف می شوند و فقط عناصر منحصر به فرد باقی می مانند.

- این ویژگی به خاطر این که set در واقع یک مجموعه (collection) از عناصر منحصر به فرد می باشد به همین دلیل برای کارهایی مثل حذف تکرارها خیلی کاربردی می باشد.

Subject: Date:

F. Stack و Queue چه تفاوتی دارند؟

- تفاوت اصلی بین Queue و Stack در نحوه اضافه و حذف کردن عناصر شون.

- Stack (پشته):

• روش کار: به صورت LIFO (Last In, First Out) عمل می کند. یعنی آخرین عنصری که اضافه میشه اولین عنصریه که حذف میشه.

• عملیات اصلی:

- push: اضافه کردن عنصر به بالای پشته

- pop: حذف و برگرداندن آخرین عنصر اضافه شده

- Queue (صف):

• روش کار: به صورت FIFO (First In, First Out) عمل می کند. یعنی اولین عنصری که اضافه میشه اولین عنصریه که حذف میشه.

• عملیات اصلی:

- enqueue: اضافه کردن عنصر به انتهای صف.

- dequeue: حذف و برگرداندن اولین عنصر اضافه شده

Subject:

Date:

F. Hash Table چیست و چه کاربردی دارد؟

- هش تیبل (Hashtable) یک ساختار داده ای بسیار کارآمد است که برای ذخیره و جستجوی داده ها به کار می رود. این ساختار داده به کمک یک تابع هش، کلید ها را به ایندکس های عددی در آرایه ای از تیبل داده ها می کند. این کار باعث می شود که جستجو، اضافه کردن و حذف داده ها به سرعت انجام شود.

- کاربردهای هش تیبل:

1. جستجوی سریع: با استفاده از هش تیبل می توان به سرعت به داده ها دسترسی پیدا کرد. زمان جستجو معمولاً O(1) است.

2. ذخیره سازی داده ها: برای ذخیره سازی مجموعه ای از داده ها با کلیدهای منحصر به فرد، مثل دیکشنری ها در زبان های برنامه نویسی.

3. مدیریت جلسات: در وب سایت ها برای ذخیره سازی اطلاعات مربوط به کاربران و مدیریت جلسات.

4. سازی داده ها: برای ذخیره سازی اطلاعات به صورت فشرده و سریع.

Date: Sat Sun Mon Tue Thu Wed Fri Subject: ----------

6. Binary Tree و B-Tree چه تفاوتی دارند؟

1. ساختار:

Binary Tree: در این درخت هر گره می تواند حداکثر دو فرزند داشته باشد (چپ و راست).

این دو ساختار ساده است و معمولاً برای جست وجو و مرتب سازی داده ها استفاده می شود.

B-Tree: این نوع درخت به گونه ای طراحی شده که می تواند به طور قابل توجهی کاهش یابد

بیش از دو فرزند داشته باشد

2. تعادل:

Binary Tree: ممکن است تعادل نداشته باشد و در نتیجه عملکرد جستجو می تواند به طور قابل

توجهی کاهش یابد (مثلاً در صورت ایجاد یک درخت خطی)

B-Tree: به طور خودکار تعادل را حفظ می کند. به این معنی است که ارتفاع درخت کم

می ماند و جستجو، درج و حذف داده ها به طور موثرتر انجام می شود.

Date: / / Sat Sun Mon Tue Thu Wed Fri Subject: ----------

3. کاربرد:

Binary Tree : بیشتر برای ساختارهای داده‌ای ساده جست‌وجوهای سریع و الگوریتم‌های مرتب‌سازی استفاده می‌شود.

B-Tree : به ویژه در سیستم‌های مدیریت پایگاه داده و سیستم‌های فایل به کار می‌رود جایی که نیاز به خواندن و نوشتن بلوک‌های بزرگ داده وجود دارد.

H. چرا Graph Data Structure برای شبکه‌های اجتماعی استفاده می‌شود؟ ساختار داده‌ای گراف به خاطر ویژگی‌های خاصش، انتخاب خیلی خوبی برای مدل‌سازی شبکه‌های اجتماعی هست.

1. ارتباطات پیچیده: در شبکه‌های اجتماعی، کاربران به همدیگه متصل هستند و این ارتباطات می‌تونند پیچیده باشند، گراف به خوبی می‌تونه روابط بین کاربران رو نشون بده.

2. رابطه‌های چندگانه: هر کاربر می‌تونه با چندین کاربر دیگر ارتباط داشته باشه، در گراف می‌تونیم روابط مختلف (دوستی، فالو کردن، عضو گروه بودن و ...) رو به راحتی مدل‌سازی کنیم.

Date: Sat Sun Mon Tue Thu Wed Fri Subject:

تحلیل شبکه: با استفاده از الگوریتم های گراف، می‌توانیم اطلاعات مفیدی استخراج کنیم، مثل پیدا کردن افراد تاثیرگذار، تحلیل گروه ها و...

4. مقیاس پذیری: گراف ها می‌توانند به راحتی مقیاس پذیر باشند و با افزایش تعداد کاربران و ارتباطات، عملکردشان را حفظ کنند.

أ. Dynamic Programming چرا در حل مسائل پیچیده کاربرد دارد؟

DP یک تکنیک قدرتمند برای حل مسائل پیچیده است، به ویژه وقتی که مسئله‌ها می‌توانند به زیرمسئله‌های کوچکتر تقسیم شوند. این روش به چند دلیل کاربردی است:

1. کاهش پیچیدگی زمانی

2. حل مسائل بهینه‌سازی

3. ساختار بهینه

4. قابلیت استفاده مجدد

Date: Sat Sun Mon Tue Thu Wed Fri Subject:

Recursion چیست و چرا در الگوریتم های پیشرفته استفاده می شود؟

Recursion (بازگشت) یک تکنیک برنامه نویسی است که در آن یک تابع خودرا فراخوانی می کند تا مسئله را به زیر مسئله های کوچکتر تقسیم کند. این روش در الگوریتم ها به دلیل سادگی و وضوح و قابلیت حل مسائل پیچیده مانند درختان و گراف ها استفاده می شود.