در فرهنگ لغت صف به معنای خط انتظار است.

مثال: صفی از کارها در سیستم کامپیوتری که منتظرند از یک دستگاه خروجی مثل چاپ گر استفاده کنند.

صف: مجموعه ای از عناصر مرتب است که هر عنصر از یک طرف به نام جلوی صف از آن حذف می شود و از طرف دیگر به ناانتهای صف در صف قرار می گیرد. به همین دلیل آن را ساختمان داده FIFO (خروج به ترتیب در مود) می نامند.

ADT صف

عناصر داده از مجموعه ای مرتب از عناصر که در آن، عناصر از یک طرف به نام جلوی صف حذف و از طرف دیگر به نام انتهای صف به آن اضافه می شوند.

عملیات اصلی:

- ایجاد صف: یک صف خالی ایجاد می کند.

- تست خالی بودن صف

- افزودن عنصر به انتهای صف

- بازیابی عنصری از جلوی صف: ولی آن را از صف حذف نمی کند.

شکل زیر را در نظر بگیرید که متغیرهای rear و frant به ترتیب آدرس های آخر و جلوی صف را نگه داری می کنند. مراحل مختلف حذف و اضافه عناصر را نشان می دهد.

شکل ص 90

ساختمان داده صف را می توان با روش های

* خطی [آرایه]
* اشاره گر [لیست پیوندی]

پیاده سازی نمود. با عنایت به مشخصات هرکدام از آن ها محدودیت هایی برای پیاده سازی ایجاد می شود. به هر حال در ادامه با شکل به توضیح و ارائه راه حل برای برخی از این محدودیت ها می پردازیم.

مشکلات پیاده سازی صف با آرایه: شکل زیر را در نظر بگیرید:

شکل ص 97

قسمت (د) را در نظر بگیرید. صف جای خالی دارد. اما مقدار rear 4 است و برای افزودن F باید مقدار آن 5 شود که خارج از حد آرایه است. پس مشکل این است که با وجود خالی بودن صف نمی توان عنصر اضافه کرد.

مشکلات پیاده سازی با آرایه هنگام حذف و اضافه کردن عنصر جدید رخ می دهد.

درهرحال صف یک ساختمان داده است مخاطب و ساختمان داده انسان است که بتواند تجسم واضح تری از چیدمان داده ها و ارتباط آن ها با هم داشته باشد.

برای رفع مشکلات مذکور می توان صف را به صورت ................ در آورد.

یعنی متغیرهای rear و frant به صورت برپا رسیدن به حد مشخص شده مثلاً 4= rear دوباره ریست شوند و از صفر شروع کنند در شکل زیر پیاده سازی صف حلقوی نشان داده شده است.

شکل ص 99

در این روش تشخیص خالی بودن صف است. شرط

Rear < front

اینجا صدق نمی کند. مثلاً شکل (ب) با وجود این که رابطه بالا صادق است ولی صف خالی نیست. بنابراین در این جا شده به صورت

Rear= = front

بیان می شود.

پر بودن صف را نیز باید بتوان چک کرد. شرط بالا می تواند بدین منظور استفاده شود. بنابراین نمی توان پر بودن و خالی بودن صف را از هم تشخیص داد.

برای رفع این مشکل صف نیمه پر را ......... در نظر می گیریم.

شکل ص 100

وقتی یک محل خالی در آرایه وجود داشته باشد، صف را به عنوان صف پر در نظر می گیریم. شرط پر بودن به صورت زیر تغییر می کند:

(rear =1) % Size = = front

در هر 2 نوع صف (خطی و حلقوی) ، تعداد عناصر موجود در صف از رابطه ی زیر بدست می آید:

Rear – front + 1

صف اولیت: در این نوع صف اضافه کردن عنصر جدید به هر ترتیبی امکان پذیر است ولی حذف یک عنصر از آن به صورت مرتب انجام می شود.

چند مثال را بررسی می کنیم.

مثال (صوف اولویت نزولی): در عمل حذف، بزرگ ترین عنصر صف حذف می شود.

مثال (صوف اولویت صعودی): در عمل حذف، کوچک ترین عنصر صف حذف می شود.

مثال: (صف سیستم عامل زمانی): آن کاری که مدت زمان بیشتری در صف انتظار، منتظر دریافت صنایع سیستم برده است.

اولویت حذف بالاتری نسبت به بقیه دارد و موقع تصمیم به حذف کردن یک عنصر زودتر از بقیه نوبت دهی از صف خارج می شود.

اصطلاحات:

Priority queue

Ascendiny priority queay

Descending priority queay

جمع بندی: مشکلات صف به طور کلی شامل موارد زیر می شود. اکثر این موارد هنگام برنامه نویسی پیش می آید. موارد دیگری نیز ممکن است پیش بینی شده اتفاق بیافتد.

1- محدودیت های آرایه: همه ی عناصر باید همگن باشند. (هم نوع هم اندازه)

2- طول آرایه از پیش تعریف شده و ثابت است. با حذف تعداد زیادی عنصر [به خصوص در صف های اولویت که ترتیب حذف مستحق نیست] با آرایه اسپارس (خلوت) مواجه می شویم. اکثر مشکلات مذکور با بهبود روش های کدنویسی و یا تغییر در ساختمان داده، راه حل پیدا کرد:

- تغییر فرم \_آرایه- تبدیل آن به صف حلقوی

- استفاده از آرایه پویا

- استفاده از لیست پیوندی

- استفاده از سخت افزارهای قوی تر (حافظه بیشتر، پردازنده پر سرعت تر، )

- تکنیک های مربوط به مدیریت بزرگ داده ها همچون پارتیشن بندی کردن

- مربوط به قسمت مشکلات ص قبل

3- به هنگام اضافه کردن عنصر در صف اولویت باید دوباره صف را مرتب سازی کنیم. مثلاً اگر اولویت صعودی بود، از کوچک تر به بزرگ تر مرتب کنیم. این طور نیست که عنصر جدید به طور خودکار (همانند صف معمولی) به آخر صف اضافه شود.

کلاس پیاده سازی صف با لیست پیوندی ص 155.