数据结构与算法（Python版）

队列抽象数据类型及Python实现

陈斌 北京大学 gischen@pku.edu.cn

队列Queue：什么是队列?

- 队列是一种有次序的数据集合，其特征是

  新数据项的添加总发生在一端（通常称为“尾rear”端）

  而现存数据项的移除总发生在另一端（通常称为“首front”端）

- 当数据项加入队列，首先出现在队尾，随着队首数据项的移除，它逐渐接近队首。

队列Queue：什么是队列?

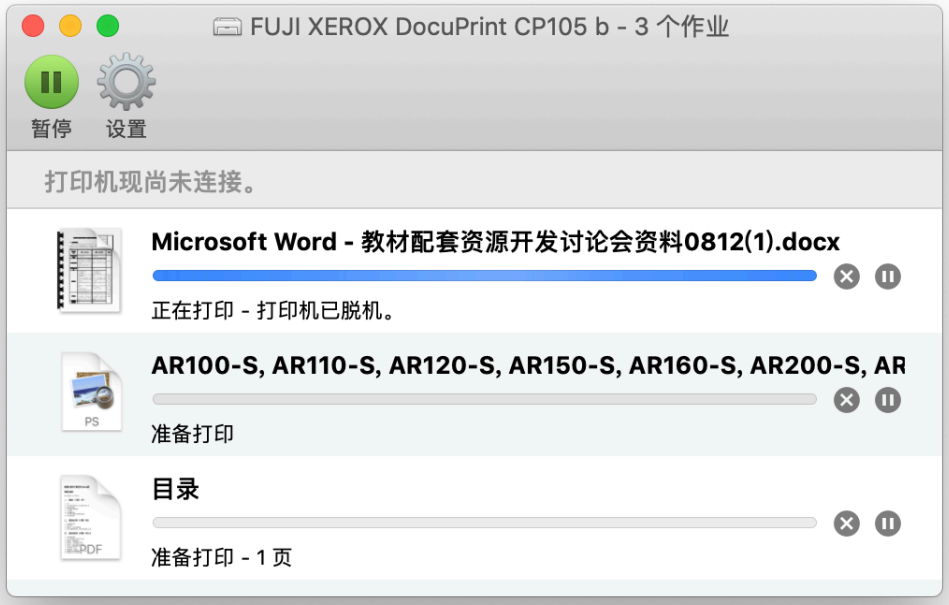
- 新加入的数据项必须在数据集末尾等待，而等待时间最长的数据项则是队首
- 这种次序安排的原则称为（FIFO:First-in first-out）先进先出
  或“先到先服务first-come first-served”
- 队列的例子出现在我们日常生活的方方面面：排队
- 队列仅有一个入口和一个出口
  不允许数据项直接插入队中，也不允许从中间移除数据项

计算机科学中队列的例子：打印队列

- 一台打印机面向多个用户/程序提供服务

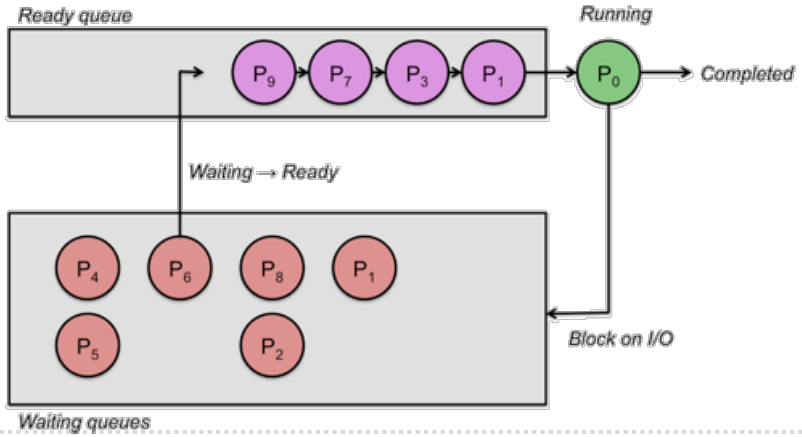
  打印速度比打印请求提交的速度要慢得多

  有任务正在打印时，后来的打印请求就要排成队列，以FIFO的形式等待被处理。



计算机科学中队列的例子：进程调度

- 操作系统核心采用多个队列来对系统中同时运行的进程进行调度
  - 进程数远多于CPU核心数
  - 有些进程还要等待不同类型I/O事件
- 调度原则综合了“先到先服务”及“资源充分利用”两个出发点

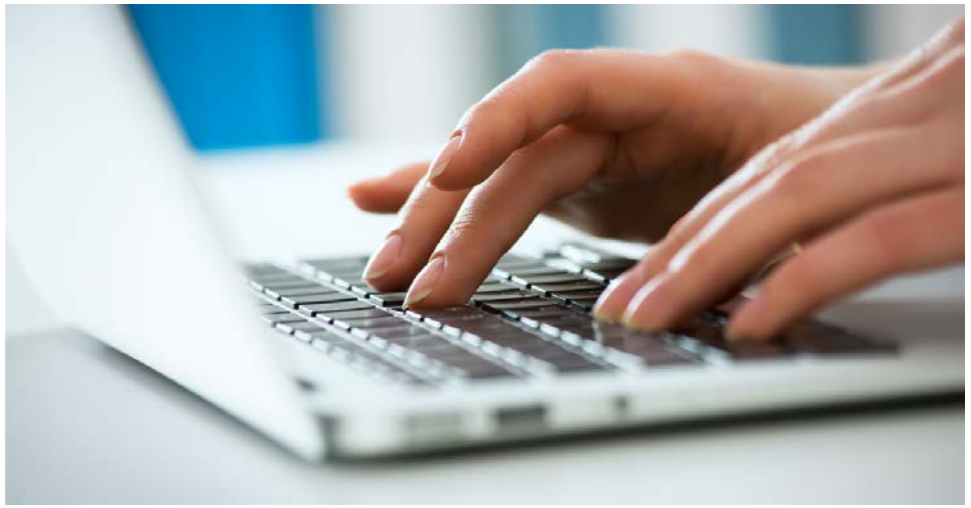


计算机科学中队列的例子：键盘缓冲

- 键盘敲击并不马上显示在屏幕上

  需要有个队列性质的缓冲区，将尚未显示的敲击字符暂存其中，

  队列的先进先出性质则保证了字符的输入和显示次序一致性。



抽象数据类型Queue

- 抽象数据类型Queue是一个有次序的数据集合
  - 数据项仅添加到“尾rear”端
  - 而且仅从“首front”端移除
  - Queue具有FIFO的操作次序

抽象数据类型Queue

- **抽象数据类型Queue由如下操作定义：**

  Queue()：创建一个空队列对象，返回值为Queue对象；

  enqueue(item)：将数据项item添加到队尾，无返回值；

  dequeue()：从队首移除数据项，返回值为队首数据项，队列被修改；

  isEmpty()：测试是否空队列，返回值为布尔值

  size()：返回队列中数据项的个数。

抽象数据类型Queue

队列操作	队列内容	返回值
q=Queue()	[]	Queue object
q.isEmpty()	[]	True
q.enqueue(4)	[4]	
q.enqueue('dog')	['dog',4]	
q.enqueue(True)	[True,'dog',4]	
q.size()	[True,'dog',4]	3
q.isEmpty()	[True,'dog',4]	False
q.enqueue(8.4)	[8.4,True,'dog',4]	
q.dequeue()	[8.4,True,'dog']	4
q.dequeue()	[8.4,True]	'dog'
q.size()	[8.4,True]	2

Python实现ADT Queue

- 采用List来容纳Queue的数据项
  - 将List首端作为队列尾端
  - List的末端作为队列首端
  - enqueue()复杂度为O(n)
  - dequeue()复杂度为O(1)
- 首尾倒过来的实现，复杂度也倒过来

```
class Queue:
    def __init__(self):
        self.items = []

    def isEmpty(self):
        return self.items == []

    def enqueue(self, item):
        self.items.insert(0,item)

    def dequeue(self):
        return self.items.pop()

    def size(self):
        return len(self.items)
```