周转时间，指的是作业从提交系统开始，直到作业完成为止的时间间隔。周转时间细分包括：

1.作业在外存中后备队列中的等待时间。

2.作业调入内存中创建的相应进程在就绪队列中的等待时间。

3.进程在cpu上执行的时间。

4.进程等待某些操作完成后的时间。

带权周转时间是指作业周转时间与作业实际运行服务时间的比值。平均周转时间和平均带权周转时间是衡量批处理系统调度算法的重要准则。

1.先来先服务调度算法：是最简单的调度算法，可以用于作业调度和进程调度。按照作业进入系统后备作业队列的先后次序来挑选作业，加入就绪队列，等待执行。

算例：假设系统中有4个作业，到达时间分别为8、8.5、9、9.5，服务时间分别为2、0.5、0.1、0.2，FCFS的调度为：

作业名	到达时间	服务时间	开始时间	完成时间	周转时间	带权周转时间
1	8	2	8	10	2	1
2	8.5	0.5	10	10.5	2	4
3	9	0.1	10.5	10.6	1.6	16
4	9.5	0.2	10.6	10.8	1.3	6.5

平均周转时间：$t=1/4(2+2+1.6+1.3)=1.725$

平均带权周转时间：$w=1/4(1+4+16+6.5)=6.875$

FCFS是非抢占式的，易于实现，效率不高，性能不好，有利于长作业而不利于短作业。

2.短作业优先调度算法，用于进程调度时又被称为短进程优先调度算法，该算法既可以用于作业调度，又可以用于进程调度。

在作业调度中，该算法每次从后备作业队列中挑选估计服务时间最短的一个或者几个作业，将他们调入内存，分配必要的资源，创建进程并放入就绪队列。在进程调度中的原理类似。

算例：假设系统中有4个作业，到达时间分别为8、8.5、9、9.5，服务时间分别为2、0.5、0.1、0.2，SJF的调度为：

作业名	到达时间	服务时间	开始时间	完成时间	周转时间	带权周转时间
1	8	2	8	10	2	1
2	8.5	0.5	10.3	10.8	2.3	4.6
3	9	0.1	10	10.1	1.1	11
4	9.5	0.2	10.1	10.3	0.8	4

平均周转时间：$t=1/4(2+2.3+1.1+0.8)=1.55$

平均带权周转时间：$w=1/4(1+4.6+11+4)=5.15$

SJF是非抢占式的，优先照顾短作业，具有很好的性能，降低平均等待时间，提高吞吐量。但是不利于长作业，长作业可能一直处于等待状态，出现饥饿现象，完全未考虑作业的优先紧迫程度，不能用于实时系统。

3.最短剩余时间优先

SJF本身是非抢占式的，用于抢占式的调度系统时，对应的算法为最短剩余作业优先调度算法。

该算法首先按照作业的服务时间挑选最短的作业运行，在该作业运行期间，一旦有新作业达到系统，并且该新作业的服务时间比当前运行作业的剩余服务时间短，则发生抢占；否则，当前作业继续运行。该算法确保一旦新的短作业或短进程进入系统，能够很快的得到处理。

算例：假设系统中有4个作业，到达时间分别为8、8.5、9、9.5，服务时间分别为2、0.5、0.1、0.2，SJF的调度为：

作业名	到达时间	服务时间	开始时间	完成时间	周转时间	带权周转时间
1	8	2	8	10.8	2.8	1.4
2	8.5	0.5	8.5	9	0.5	1
3	9	0.1	9	9.1	0.1	1
4	9.5	0.2	9.5	9.7	0.2	1

平均周转时间：t=1/4(2.8+0.5+0.1+0.2)=0.9

平均带权周转时间：w=1/4(1.4+1+1+1)=1.1

由于频繁的抢占和进程切换，系统开销大，该算法实现代价高，一般用于实时系统。

4.时间片轮转。

用于分时系统的进程调度。

基本思想：系统将CPU处理时间划分为若干个时间片，进程按照达到先后顺序排序。每次调度选择队首的进程，执行完一个时间片后，计时器发出时钟中断请求，该进程移动到队尾。以后每次调度都是如此。该算法能在给定的时间内响应所有用户的请求，达到分时系统的目的。

其性能主要取决于时间片的大小。时间片较大，则所有进程在一个时间片内完成，退化为FCFS，太小则进程频繁切换，系统开销大。

例如，假设有A、B、C、D、E五个进程，其到达系统的时间分别为0、1、2、3、4，要求运行时间依次为3、6、4、5、2，采用时间片轮转调度算法，当时间片大小分别为1和4时，试计算其平均周转时间和平均带权周转时间。

此例的最关键的问题是确定就绪队列中的进程的次序，以下分别分析当时间片为1和4时，在某些特殊时刻，就绪队列中的进程次序及正在占用CPU的进程。

(1)当时间片为1时：

0：A运行；
1：B运行，A等待；
2：A运行，CB等待；
3：C运行，BDA等待；
4：B运行，DAEC等待；
5：D运行，AECB等待；
6：A运行，ECBD等待；
7：E运行，CBD等待；
8：C运行，BDE等待；
9：B运行，DEC等待；
10：D运行，ECB等待；
11：E运行，CBD等待；
12：C运行，BD等待；
13：B运行，DC等待；
14：D运行，CB等待；
15：C运行，BD等待；
16：B运行，D等待；
17：D运行，B等待；
18：B运行，D等待；
19：D运行；
20：D结束。

其平均周转时间和平均带权周转时间如表6-5所示。

进程名	到达时间	服务时间	开始时间	完成时间	周转时间	带权周转时间
A	0	3	0	7	7	2.33
B	1	6	1	19	18	3
C	2	4	3	16	14	3.5
D	3	5	5	20	17	3.4
E	4	2	7	12	8	4

平均周转时间：t=1/5(7+18+14+17+8)=12.8

平均带权周转时间：w=1/5(2.33+3+3.5+3.4+4)=3.246

(2)当时间片为4时：

0：A运行，BCD依次到达；
3：B运行，CD等待，后E到达；
7：C运行，DEB等待；
11：D运行，EB等待；
15：E运行，BD等待；
17：B运行，D等待；
19：D运行；
20：D结束。

其平均周转时间和平均带权周转时间如表6-6所示。

进程名	到达时间	服务时间	开始时间	完成时间	周转时间	带权周转时间
A	0	3	0	3	3	1
B	1	6	3	19	18	3
C	2	4	7	11	9	2.25
D	3	5	11	20	17	3.4
E	4	2	15	17	13	6.5

平均周转时间：t=1/5(3+18+9+17+13)=12

平均带权周转时间：w=1/5(1+3+2.25+3.4+6.5)=3.23

该算法简单有效，常用于分时系统，但不利于I/O频繁的，由于这种进程用不完一个时间片，就因为等待I/O操作而被阻塞，当I/O操作结束后，只能插入到就绪队列的末尾，等待下一轮调度