全国计算机等级考试 2006

笔试题分类精解与应试策略

二级Visual FoxPro数据库程序设计

全国计算机等级考试命题研究组 编

NCR Examination 全国计算机等级考试

南開大學出版社

全国计算机等级考试

笔试题分类精解与应试策略

二级 *Visual FoxPro* 数据库程序设计

全国计算机等级考试命题研究组　编

南开大学出版社
天　津

内容提要

本书主要内容有：① 第1章~第6章，介绍二级 Visual FoxPro 等级考试的各种相关知识，通过知识点的扼要列举、真题及典型题的分类精解以及举一反三的相关练习题，使考生能够针对考试所涉及的每个知识点，逐个理解透彻，切实掌握；② 第7章应试策略，详细列出备考本科目所应具备的知识，历届考试各个知识点所占的分值范围，给出复习和考试秘笈及报考和考试注意事项。③ 最后是两套全真模拟试卷和2005年9月笔试真题试卷，并附有参考答案；④ 配套光盘中有二级 Visual FoxPro 笔试和上机考试模拟系统，提供大量模拟练习题，供考生考前自测，检验实际水平，以及熟悉考试系统的使用。

本书完全针对准备参加全国计算机等级考试二级 Visual FoxPro 数据库程序设计笔试考试的考生，同时也可以作为普通高校、大专院校、成人高等教育以及相关培训班的练习题和考试题使用。

另，备考二级 Visual FoxPro 数据库程序设计上机考试的考生，可选购本书的姊妹篇，南开大学出版社出版的《上机题分类精解与应试策略二级 Visual FoxPro 数据库程序设计》。

图书在版编目(CIP)数据

全国计算机等级考试笔试题分类精解与应试策略. 二级 Visual FoxPro 数据库程序设计 / 全国计算机等级考试命题研究组编. —天津：南开大学出版社，2006.3

ISBN 7-310-02515-6

Ⅰ.全... Ⅱ.全... Ⅲ.①电子计算机—水平考试—自学参考资料②关系数据库—数据库管理系统，Visual Foxpro—水平考试—自学参考资料 Ⅳ.TP3

中国版本图书馆 CIP 数据核字(2005)第160816号

南开大学出版社出版发行

出版人：肖占鹏

地址：天津市南开区卫津路94号 邮政编码：300071

营销部电话：(022)23508339 23500755

营销部传真：(022)23508542 邮购部电话：(022)23502200

*

天津市蓟县宏图印务有限公司印刷

全国各地新华书店经销

*

2006年3月第1版 2006年3月第1次印刷

787×1092毫米 16开本 18.25印张 448千字

定价：34.00元

如遇图书印装质量问题，请与本社营销部联系调换，电话：(022)23507125

前　言

教育部考试中心推出的全国计算机等级考试，是国内除升学考试之外参与人数最多的考试之一，具有相当的权威性、科学性和公平性，它于 1994 年推出，历经 10 年发展，已成为我国普及计算机教育不可或缺的组成部分，到目前为止，该考试已经举行过 20 余次，考生累计已逾千万。计算机等级考试的考试大纲，根据科学技术的发展和社会需求的变化，对全国计算机等级考试的科目、考核内容和考试形式多次进行调整，2004 年，推出了等级考试的第 5 个考试大纲，即 2004 年版的《考试大纲》。

本书根据教育部考试中心制定的最新考试大纲要求编写，书中覆盖了该门课程在大纲中所提到的所有内容。我们在编写本书的同时，考虑到考生参加等级考试的需要，把各种题型和训练融会在每本书之中，以期给考生带来切实的帮助。

本书的主要特点是针对性强。我们认为，在考试辅导书中，面面俱到并非是一个优势，针对性强才真正对考生有益。本书只针对等级考试二级 Visual FoxPro 数据库程序设计上机考试，不涉及无关内容。我们所分析的题目，都取自考试题库。本书主要内容如下：

① 第 1 章~第 6 章介绍二级公共基础以及 Visual FoxPro 数据库程序设计语言程序设计的各种知识，逐个知识点进行详细讲解。每一章的构成大致有 3 个方面：知识点部分，这是基础，是理解题意和正确作答的关键；“分类精解”针对题库中的典型题目进行细致透彻的解答分析，由此，考生遇到同类问题，便可以迎刃而解；“举一反三”给出相关类型题目的练习题以及答案，让考生对这方面的知识点真正掌握熟练。

② 最后一章是应试策略，本章内容不多，却是本书的精华所在，详细列出备考本科目所应具备的知识，历届考试各个知识点所占的分值范围，给出复习和考试秘笈及报考和考试注意事项。

③ 两套全真模拟试卷和 2005 年 9 月的笔试试卷及参考答案。通过这些试卷，可以熟悉考试要点、试题类型，进一步提高应试能力。

④ 配套光盘。配套光盘中有二级 Visual FoxPro 笔试和上机考试模拟系统，提供大量模拟练习题，供考生考前自测，检验实际水平，以及熟悉考试系统的使用。

为了保证本书及时面市和内容准确，很多朋友做出了贡献，齐惠颖、任世华、田民、廖明武、于樊鹏、许伟、侯佳宜、何雄、赵晓睿、戴文雅、戴军、黄志雄、李志云、陈安南、李晓春、王春桥、王雷、韦笑、龚亚萍等老师在编写文档、调试程序、排版、查错、预读、光盘制作与测试等工作中加班加点，付出了很多辛苦，在此一并表示感谢！

在学习的过程中，您如有问题或建议，请与我们联系：book_service@126.com。或登录百分网查找信息和寻求帮助：www.baifen100.com。

全国计算机等级考试命题研究组

2005 年 12 月

目　录

第 1 章　公共基础知识

MEMO

考纲要求

一、基本数据结构与算法

1. 算法的基本概念；算法复杂度的概念和意义（时间复杂度与空间复杂度）。

2. 数据结构的定义；数据的逻辑结构与存储结构；数据结构的图形表示；线性结构与非线性结构的概念。

3. 线性表的定义；线性表的顺序存储结构及其插入与删除运算。

4. 栈和队列的定义；栈和队列的顺序存储结构及其基本运算。

5. 线性单链表，双向链表与循环链表的结构及其基本运算。

6. 树的基本概念；二叉树的定义及其存储结构；二叉树的前序、中序和后序遍历。

7. 顺序查找与二分查找算法；基本排序算法（交换类排序、选择类排序、插入类排序）。

二、程序设计基础

1. 程序设计方法与风格。

2. 结构化程序设计。

3. 面向对象的程序设计方法，掌握并理解对象、方法、属性以及继承与多态性的概念。

三、软件工程基础

1. 软件工程基本概念；软件生命周期概念；软件工具与软件开发环境。

2. 结构化分析方法，数据流图，数据字典，软件需求规格说明书。

3. 结构化设计方法，总体设计与详细设计。

4. 软件测试方法，白盒测试与黑盒测试，测试用例设计，软件测试的实施，单元测试、集成测试和系统测试。

5. 程序调试，静态调试与动态调试。

四、数据库设计基础

1. 数据库的基本概念：数据库、数据库管理系统、数据库系统。

2. 数据模型，实体联系模型及 E-R 图，从 E-R 图导出关系数据模型。

3. 关系代数运算，包括集合运算及选择、投影、连接运算；数据库规范化理论。

4. 数据库设计方法和步骤：需求分析、概念设计、逻辑设计和物理设计的相关策略。

1.1 数据结构与算法

根据历届考试的真题分析，本知识点分值分配：10~14分。

1.1.1 算法

1. 算法的基本概念

所谓算法是指解题方案的准确而完整的描述。

对于一个问题，如果可以通过一个计算机程序，在有限的存储空间内运行有限长的时间而得到正确的结果，则称这个问题是算法可解的。但算法不等于程序，也不等于计算方法。当然，程序也可以作为算法的一种描述，但程序通常还需考虑很多与方法和分析无关的细节问题，这是因为在编写程序时要受到计算机系统运行环境的限制。

算法实际上是一种抽象的解题方法，它具有动态性。作为一个算法，一般应具有以下几个基本特征。

（1）可行性

算法的可行性主要包括两个方面。一是算法中的每一个步骤必须是能实现的。二是算法执行的结果要能达到预期的目的。

（2）确定性

算法的确定性，是指算法中的每一个步骤都必须是有明确定义的，不允许有模凌两可的解释，也不允许有多义性。

（3）有穷性

算法的有穷性，是指算法必须能在有限的时间内做完，即算法必须能在执行有限个步骤之后终止。

算法的有穷性还应包括合理的执行时间的含义。因为，如果一个算法需要执行千万年，也失去了实用价值。

（4）拥有足够的情报

一个算法是否有效，还取决于为算法所提供的情报是否足够。通常，算法中的各种运算总是要施加到各个运算对象上，而这些运算对象又可能具有某种初始状态，这是算法执行的起点或是依据。因此，一个算法执行的结果总是与输入的初始数据有关，不同的输入将会有不同的结果输出。当输入不够或输入错误时，算法本身也就无法执行或导致执行有错。一般来说，当算法拥有足够的情报时，此算法才是有效的，而当提供的情报不够时，算法并不有效。

综上所述，所谓算法，是一组严谨地定义运算顺序的规则，并且每一个规则都是有效的，且是明确的，此顺序将在有限的次数下终止。

2. 算法复杂度

算法的复杂度主要包括时间复杂度和空间复杂度。

（1）算法的时间复杂度

算法的时间复杂度是指执行算法所需要的计算工作量。

算法的工作量用算法所执行的基本运算次数来度量，而算法所执行的基本运算次数是问题规模的函数，即

算法的工作量＝f(n)

其中n是问题的规模。

在同一问题规模下，如果算法执行所需的基本运算次数取决于某一特定输入时，可以用以下两种方法来分析算法的工作量。

平均性态分析，是指用各种特定输入下的基本运算次数的带权平均值来度量算法的工作量。算法的平均性态定义为（　　）。

$$A(n)=\sum_{x\in D_n}p(x)t(x)$$

其中：x是所有可能输入中的某个特定输入，p（x）是x出现的概率（即输入为x的概率），t（x）是算法在输入为x时所执行的基本运算次数，D_n表示当规模为n时，算法执行时所有可能输入的集合。

最坏情况分析，是指在规模为 n 时，算法所执行的基本运算的最大次数。它定义为（　　）。

$$W(n)=\max_{x\in D_n}\{t(x)\}$$

显然，W（n）的计算要比A（n）的计算方便得多。由于W（n）实际上是给出了算法工作量的一个上界，因此，它比A（n）更具有实用价值。

（2）算法的空间复杂度

一个算法的空间复杂度，一般是指执行这个算法所需要的内存空间。

一个算法所占用的存储空间包括算法程序所占的空间、输入的初始数据所占的存储空间以及算法执行过程中所需要的额外空间。其中额外空间包括算法程序执行过程中的工作单元以及某种数据结构所需要的附加存储空间。如果额外空间量相对于问题规模来说是常数，则称该算法是原地（in place）工作的。在许多实际问题中，为了减少算法所占的存储空间，通常采用压缩存储技术，以便尽量减少不必要的额外空间。

分　类　精　解

【例1-1】

下面叙述正确的是（　　）。

A）算法的执行效率与数据的存储结构无关

B）算法的空间复杂度是指算法程序中指令（或语句）的条数

C）算法的有穷性是指算法必须能在执行有限个步骤之后终止

D）以上三种描述都不对

【解析】A选项错误，因为算法的执行效率与算法执行过程中所需基本运算的执行次数有关；B选项错误，原因是算法的空间复杂度是指执行这个算法所需要的内存空间；C选项正确，故D选项不正确。

因此，本题的正确答案为C。

【例 1-2】

算法的复杂度主要包括____复杂度和空间复杂度。

【解析】算法的复杂度主要指时间复杂度和空间复杂度。所谓算法的时间复杂度，是指执行算法所需要的计算工作量；算法的空间复杂度，一般是指执行这个算法所需要的内存空间。

因此，本题的正确答案为时间。

【例 1-3】问题处理方案的正确而完整的描述称为____ 。

【解析】算法是问题处理方案正确而完整的描述。

因此，本题的正确答案为算法　或　程序　或　流程图。

举　一　反　三

1. 以下内容不属于算法程序所占的存储空间的是（　　）。

A）算法程序所占的空间

B）输入的初始数据所占的存储空间

C）算法程序执行过程中所需要的额外空间

D）算法执行过程中所需要的存储空间

2. 以下特点不属于算法的基本特征的是（　　）。

A）可行性　　B）确定性

C）无穷性　　D）拥有足够的情报

3. 下面叙述正确的是（　　）。

A）算法的执行效率与数据的存储结构无关

B）算法的空间复杂度是指算法程序中指令（或语句）的条数

C）算法的有穷性是指算法必须能在执行有限个步骤之后终止

D）以上三种描述都不对

4. 算法的时间复杂度是指（　　）。

A）执行算法程序所需要的时间

B）算法程序的长度

C）算法执行过程中所需要的基本运算次数

D）算法程序中的指令条数

5. 算法的空间复杂度是指（　　）。

A）算法程序的长度

B）算法程序中的指令条数

C）算法程序所占的存储空间

D）算法执行过程中所需要的存储空间

6. 算法的复杂度主要包括____复杂度和空间复杂度。

【答案】

1. D　2. C　3. C　4. C　5. D　6. 时间

1.1.2 数据结构

1. 数据结构的基本概念

数据结构是指反映数据元素之间关系的数据元素集合的表示。通俗地说，数据结构是指带有结构的数据元素的集合。

（1）数据的逻辑结构

一般情况下，在具有相同特征的数据元素集合中，各个数据元素之间存在有某种关系，这种关系反映了该集合中的数据元素所固有的一种结构。在数据处理领域中，通常把数据之间这种固有的关系简单地用前后件关系（或直接前驱与直接后继关系）来描述。

所谓数据的逻辑结构，是指反映数据元素之间逻辑关系的数据结构。

数据的逻辑结构有两个要素：一是数据元素的集合，通常记为D；二是D上的关系，它反映了D中各数据元素之间的前后件关系，通常记为R。即一个数据结构可以表示成

```
B=(R,D)
```

其中B表示数据结构。为了反映D中各数据元素之间的前后件关系，一般用二元组来表示。例如，假设a与b是D中的两个数据，则二元组（a，b）表示a是b的前件，b是a的后件。这样，在D中的每两个元素之间的关系都可以用这种二元组来表示。

例如，一年四季的数据结构可以表示成

```
B=(R,D)
D={春，夏，秋，冬}
R={（春，夏），（夏，秋），（秋，冬）}
```

家庭成员数据结构可以表示成

```
B=(R,D)
D={父亲，儿子，女儿}
R={（父亲，儿子），（父亲，女儿）}
```

（2）数据的存储结构

各数据元素在计算机存储空间中的位置关系与它们的逻辑关系不一定是相同的，而且一般也不可能相同。例如在一年四季的数据结构中，“春”是“夏”的前件，但在对它们进行处理时，在计算机存储空间中，“春”这个数据元素的信息不一定被存储在“夏”这个数据元素信息的前面，而可能在后面，也可能不是紧邻在前面，而是中间被其他的信息所隔开。

数据的逻辑结构在计算机存储空间中的存放形式称为数据的存储结构（也称数据的物理结构）。

由于数据元素在计算机存储空间中的位置关系可能与逻辑关系不同，因此，为了表示存放在计算机存储空间中的各数据元素之间的逻辑关系（即前后件关系），在数据的存储结构中，不仅要存放各数据元素的信息，还需要存放各数据元素之间的前后件关系的信息。

一般来说，一种数据的逻辑结构根据需要可以表示成多种存储结构，常用的存储结构有顺序、链接、索引等存储结构。采用不同的存储结构，其数据处理的效率是不同的。

2. 数据结构的图形表示

一个数据结构除了用二元关系表示外，还可以直观地用图形表示。在数据结构的图形表示中，对于数据集合 D 中的每一个数据元素用中间标有元素值的方框表示，一般称之为数据结点，并简称为结点；为了进一步表示各数据元素之间的前后件关系，对于关系 R 中的每一个二元组，用一条有向线段从前件结点指向后件结点。

例如，一年四季的数据结构可以用图 1.1 所示的图形来表示。

图 1.1　一年四季数据结构的图形表示

又如，反映家庭成员间辈分关系的数据结构可以用图 1.2 所示的图形来表示。

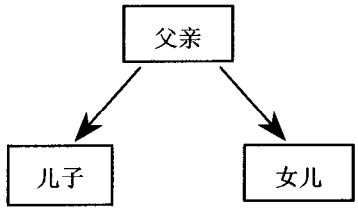

图 1.2　家庭成员数据结构的图形表示

3. 线性结构与非线性结构

根据数据结构中个数据元素之间前后件关系的复杂程度，一般将数据结构分为两大类型：线性结构与非线性结构。

如果一个非空的数据结构满足下列两个条件：

① 有且只有一个根结点。

② 每一个结点最多有一个前件，也最多有一个后件。

则称该数据结构为线性结构。线性结构又称线性表。

如果一个数据结构不是线性结构，则称之为非线性结构。如一年四季这个数据结构。

需要特别说明的是：在一个线性结构中插入或删除任何一个结点后还应是线形结构。

如果一个数据结构不是线性结构，则称之为非线性结构。如反映家庭成员间辈分关系的数据结构属于非线性结构。

如果在一个数据结构中一个数据元素都没有，则称该数据结构为空的数据结构。线性结构和非线性结构都可以是空的数据结构。如果对一个空数据结构的运算是按线性结构的规则来处理，则属于线性结构；否则属于非线性结构。

4. 线性表及其存储结构

（1）线性表的基本概念

线性表是由 n（n≥0）个数据元素 a1，a2，…，an 组成的一个有限序列，表中的每一个数据元素，除了第一个外，有且只有一个前件，除了最后一个外，有且只有一个后件。即线性表或是一个空表，或可以表示为：

(a1, a2, …, ai, …, an)

其中 a_i(i=1, 2, …, n)是属于数据对象的元素，通常也称其为线性表中的一个结点。

非空线性表有如下一些结构特征：

① 有且只有一个根结点 a1，它无前件。

② 有且只有一个终端结点 an，它无后件。

③ 除根结点与终端结点外，其他所有结点有且只有一个前件，也有且只有一个后件。线性表中结点的个数 n 称为线性表的长度。当 n=0 时，称为空表。

（1）线性表的顺序存储结构

线性表的顺序存储结构具有以下两个基本特点：

① 线性表中所有元素所占的存储空间是连续的。

② 线性表中各数据元素在存储空间中是按逻辑顺序依次存放的。

在线性表的顺序存储结构中，其前后件两个元素在存储空间中是紧邻的，且前件元素一定存储在后件元素的前面。

在顺序存储结构中，线性表中每一个数据元素在计算机存储空间中的存储地址由该元素在线性表中的位置序号唯一确定。

在程序设计语言中，通常定义一个一维数组来表示线性表的顺序存储空间。

（3）顺序表的插入运算

设长度为 n 的线性表为（　　）。

$$(a_1,a_2,\cdots,a_i,\cdots,a_n)$$

要在线性表的第 i 个元素 ai 之前插入一个新元素 b，插入后得到长度为 n+1 的线性表为（　　）。

$$(a'_1,a'_2,\cdots,a'_j,a'_{j+1},\cdots,a'_n,a'_{n+1})$$

则插入前后的两线性表中的元素满足如下关系：

$$a'_j=\begin{cases} a_j & 1\le j\le i-1 \\ b & j=i \\ a_{j-1} & i+1\le j\le n+1 \end{cases}$$

在一般情况下，要在第 i（1≤i≤n）个元素之前插入一个新元素时，首先要从最后一个（即第 n 个）元素开始，直到第 i 个元素之间共 n−i+1 个元素依次向后移动一个位置，移动结束后，第 i 个位置就被空出，然后将新元素插入到第 i 项。插入结束后，线性表的长度就增加了 1。

显然，在线性表采用顺序存储结构时，如果插入运算在线性表的末尾进行，即在第 n 个元素之后（可以认为是在第 n+1 个元素之前）插入新元素，则只要在表的末尾增加一个元素即可，不需要移动表中的元素；但如果要在线性表的第 1 个元素之前插入一个新元素，则需要移动表中所有的元素。在一般情况下，如果插入运算在第 i（1≤i≤n）个元素之前进行，则原来第 i 个元素之后(包括第 i 个元素)的所有元素都必须移动。在平均情况下，要在线性表中插入一个新元素，需要移动表中一半的元素。因此，在线性表顺序存储的情况下，要插入一个新元素，其效率是很低的，特别是在线性表比较大的情况下更为突出，由

于数据元素的移动而消耗较多的处理时间。

（4）顺序表的删除运算

设长度为 n 的线性表为（　　）。

$(a_1,a_2,\cdots,a_i,\cdots,a_n)$

现要删除第 i 个元素，删除后得到长度为 n－1 的线性表为（　　）。

$(a'_1,a'_2,\cdots,a'_j,\cdots,a'_{n-1})$

则删除前后的两线性表中的元素满足如下关系：

$$a'_j=\begin{cases}a_j & 1\le j\le i-1\\ b & j=i\\ a_{j-1} & i+1\le j\le n+1\end{cases}$$

在一般情况下，要删除第 i（1≤i≤n）个元素时，则要从第 i＋1 个元素开始，直到第 n 个元素之间共 n－i 个元素依次向前移动一个位置。删除结束后，线性表的长度就减小了 1。

显然，在线性表采用顺序存储结构时，如果删除运算在线性表的末尾进行，即删除第 n 个元素，则不需要移动表中的元素；但如果要删除线性表中的第 1 个元素，则需要移动表中所有的元素。在一般情况下，如果要删除第 i（1≤i≤n）个元素，则原来第 i 个元素之后的所有元素都必须依次往前移动一个位置。在平均情况下，要在线性表中删除一个元素，需要移动表中一半的元素。因此，在线性表顺序存储的情况下，要删除一个元素，其效率也是很低的，特别是在线性表比较大的情况下更为突出，由于数据元素的移动而消耗较多的处理时间。

5. 栈

（1）栈的概念

栈（stack）是限定在一端进行插入与删除的线性表。在这种特殊的线性表中，其插入与删除运算都只在线性表的一端进行。即在这种线性表的结构中，一端是封闭的，不允许进行插入与删除元素；另一端是开口的，允许插入与删除元素。

在栈中，允许插入与删除的一端称为栈顶，而不允许插入与删除的另一端称为栈底。栈顶元素总是最后被插入的元素，最先删除的元素；栈底元素总是最先被插入的元素，最后被删除的元素。栈是按照“先进后出”（FILO—First In Last Out）或“后进先出”（LIFO—Last In First Out）的原则组织数据的，因此，栈也被称为“先进后出”表或“后进先出”表。由此可以看出，栈具有记忆作用。

通常用指针 top 来指示栈顶的位置，用指针 bottom 指向栈底。往栈中插入一个元素称为入栈运算，从栈中删除一个元素（即删除栈顶元素）称为退栈运算。栈顶指针 top 动态反映了栈中元素的变化情况。

（2）栈的顺序存储及其运算

在程序设计语言中，用一维数组 S（1：m）作为栈的顺序存储空间，其中 m 为栈的最大容量。在栈的顺序存储空间 S（1：m）中，S（bottom）通常为栈底元素（在栈非空的情况下），S（top）为栈顶元素。top＝0 表示栈空；top＝m 表示栈满。

入栈运算。入栈运算是指在栈顶位置插入一个新元素。这个运算有两个基本操作：首先将栈顶指针进一（即 top 加 1），然后将新元素插入到栈顶指针指向的位置。

退栈运算。退栈运算是指取出栈顶元素并赋给一个指定的变量。这个运算有两个基本操作：首先将栈顶元素（栈顶指针指向的元素）赋给一个指定的变量，然后将栈顶指针退一（即 top 减 1）。当栈顶指针为 0 时，说明栈空，不可能进行退栈操作。这种情况称为栈“下溢”错误。

读栈顶元素。读栈顶元素是指将栈顶元素赋给一个指定的变量。这个运算不删除栈顶元素，只是将它的值赋给一个变量，因此，在这个运算中，栈顶指针不会改变。

6. 队列

（1）队列的概念

队列（equeue）是指允许在一端进行插入、而在另一端进行删除的线性表。允许插入的一端称为队尾，通常用一个称为尾指针（rear）的指针指向队尾元素，即尾指针总是指向最后被插入的元素；允许删除的一端称为排头（也称为队头），通常也用一个排头指针（front）指向排头元素的前一个位置。显然，在队列这种数据结构中，最先插入的元素将最先被删除，反之，最后插入的元素将最后才能被删除。因此，队列又称为“先进先出”（FIFO—First In First Out）或“后进后出”（LILO—Last In Last Out）的线性表，它体现了“先来先服务”的原则。在队列中，队尾指针 rear 与排头指针 front 共同反映了队列中元素动态变化的情况。

（2）循环队列及其运算

所谓循环队列，就是将队列存储空间的最后一个位置绕到第一个位置，形成逻辑上的环状空间，供队列循环使用。在循环队列结构中，当存储空间的最后一个位置已被使用而再要进行入队运算时，只要存储空间的第一个位置空闲，便可将元素加入到第一个位置，即将存储空间的第一个位置作为队尾。

在循环队列中，用队尾指针 rear 指向队列中的队尾元素，用排头指针 front 指向排头元素的前一个位置，因此，从排头指针 front 指向的后一个位置直到队尾指针 rear 指向的位置之间所有的元素均为队列中的元素。

循环队列的初始状态为空，即 rear＝front＝m。

循环队列主要有两种基本运算：入队运算与退队运算。

每进行一次入队运算，队尾指针就进一。当队尾指针 rear＝m＋1 时，则置 rear＝1。

每进行一次退队运算，排头指针就进一。当排头指针 front＝m＋1 时，则置 front＝1。

在实际使用循环队列时，为了能区分队列满还是队列空，通常还需增加一个标志 s，s 值的定义如下：

$$s=\begin{cases}0 & \text{表示队列空}\\ 1 & \text{表示队列非空}\end{cases}$$

由此可以得出队列空与队列满的条件如下：

① 队列空的条件为 s＝0。

② 队列满的条件为 （s＝1）且（front＝rear）。

循环队列入队与退队的算法如下。