第3章练习题

**一. 单选题（共 27 分）**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. | 一栈的入栈序列是 a, b, c, d, e，则栈的不可能的输出序列是（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | e,d,c,b,a | B. | d,e,c,b,a | | C. | d,c,e,a,b | D. | a,b,c,d,e | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2. | 在一个具有 n 个单元的顺序栈中，假定以地址低端（即下标为 0 的单元）作为栈底，以 top 表示栈顶元素的下标，当入栈时， top 的变化应为（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 不变 | B. | top = 0; | | C. | top--; | D. | top++; | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3. | 栈是一种特殊的线性表，其特殊性体现在（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 数据元素的类型是一个字符 | B. | 使用了顺序存储结构 | | C. | 只能在表的一端进入插入，在另一端进行删除 | D. | 只能在固定在表的一端进行插入和删除操作，另一端不能进行操作。 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 4. | 队列是一种特殊的线性表，其特殊性体现在（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 数据元素的类型是一个字符 | B. | 使用了顺序存储结构 | | C. | 只能固定在表的一端进行插入，在另一端进行删除 | D. | 只能在固定在表的一端进行插入和删除操作，另一端不能进行操作。 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 5. | 循环队列使用长度数组 N 的数组来存储数据元素，为保证能正确地判断出队列空或队列满，浪费一个数组单元，使用front来标识队头元素的“前”一个位置，用rear来标识队尾元素。该循环队列的有效元素个数为（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | (rear - front + N) % N | B. | (rear - front) % N | | C. | (front - rear + N) % N | D. | (front - rear) % N | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 6. | 循环队列使用长度数组 N 的数组来存储数据元素，为保证能正确地判断出队列空或队列满，浪费一个数组单元，使用front来标识队首的“前”一个元素，用rear来标识队尾元素。则队列满的条件是（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | (rear + 1) % N == front | B. | (front + 1) % N == rear | | C. | (rear - 1) % N == front | D. | front == rear | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7. | 已知SeqStack是顺序栈类模板，执行以下代码后，输出为（ ）。 （1 分） 注：Push为入栈操作函数，Pop为出栈操作函数，GetTop为取栈顶元素  SeqStack<char> s; s.Push('a'); s.Push('b'); s.Pop(); cout << s.GetTop();   |  |  |  |  | | --- | --- | --- | --- | | A. | a | B. | b | | C. | 1 | D. | 2 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 8. | 链栈与顺序栈相比，比较明显的优点是（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 插入操作更加方便 | B. | 删除操作更加方便 | | C. | 不会出现下溢的情况 | D. | 不会出现上溢的情况 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 9. | 在一个具有 n 个单元的顺序栈中，假定以地址低端（即下标为 0 的单元）作为栈底，以 top 作为栈顶指针，当出栈时，top 的变化为（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 不变 | B. | top = 0; | | C. | top = top - 1; | D. | top = top + 1 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10. | 从栈顶指针为 top 的链栈中删除一个结点，用 x 保存被删除结点的数据，则执行（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | x = top; top = top->next; | B. | x = top->data; | | C. | top = top->next; x = top->data; | D. | x = top->data; top = top->next; | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 11. | 循环队列的队头和队尾指针分别为front和rear，则判断循环队列为空的条件是（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | front==rear | B. | front==0 | | C. | rear==0 | D. | front=rear+1 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 12. | 在一个链队列中，front和rear分别为头指针和尾指针，则插入一个结点s的操作为（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | front=front->next; | B. | s->next=rear;rear=s; | | C. | rear->next=s;rear=s; | D. | s->next=front;front=s; | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 13. | 在解决计算机主机与打印机之间速度不匹配问题时，通常设置一个打印数据缓冲区，主机将要输出的数据依次写入该缓冲区，而打印机则从该缓冲区中取走数据打印。该缓冲区应该是一个（ ）结构。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 栈 | B. | 队列 | | C. | 数组 | D. | 线性表 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 14. | 在一个链队列中，假定front和rear分别为队头和队尾指针，删除一个结点的操作是（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | front=front->next; | B. | rear=rear->next; | | C. | rear->next=rear; | D. | front->next=front; | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 15. | 多项式表达式求值是（ ）的典型实例。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 队列 | B. | 栈 | | C. | 图 | D. | 树 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 16. | 若栈采用顺序存储方式存储，现两栈共享空间V[1……m],top(i)代表第i个栈（i=1，2）栈顶，栈1的底在V[1]，栈2的底在V[m]，则栈满的条件是（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | top[2]-top[1]=0 | B. | top[1]+1=top[2] | | C. | top[1]+top[2]=m | D. | top[1]=top[2] | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 17. | 以下（ ）不是队列的基本运算。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 在队列第i个元素之后插入一个元素 | B. | 从队头删除一个元素 | | C. | 判断一个队列是是否为空 | D. | 读取队头元素的值 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 18. | 若元素a、b、c、d、e、f依次进栈，允许进栈、出栈操作交替进行，但不允许连续三次进行出栈操作，则不可能得到的出栈序列是( )。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | d c e b f a | B. | b c a e f d | | C. | c b d a e f | D. | a f e d c b | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 19. | 元素a，b，c，d，e依次进入初始为空的栈中，若元素进栈后可停留、可出栈，直到所有元素都出栈，则在所有可能的出栈序列中，以元素d开头的序列个数是（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 3 | B. | 4 | | C. | 5 | D. | 6 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 20. | 若用一个大小为 5 的数组来实现循环队列，且当前rear和front的值分别为 0 和 2 当从队列中删除 2 个元素，再加入 1 个元素后，rear和front的值分别是（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 2 3 | B. | 1 4 | | C. | 4 1 | D. | 3 2 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 21. | 若已知一个栈的入栈序列是 1, 2, 3,...n，其输出序列为p1,p2,p3,..,pn,若p1=n,则pi为（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | i | B. | n-i | | C. | n-i+1 | D. | 不确定 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 22. | 一栈的入栈序列是 1, 2, 3, 4,5，则栈的不可能的输出序列是（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 5，4，3，2，1 | B. | 2，1，5，4，3 | | C. | 4，3，1，2，5 | D. | 2，3，5，4，1 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 23. | 模拟打印机缓冲区是（ ）的典型实例。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 队列 | B. | 栈 | | C. | 图 | D. | 树 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 24. | 链栈结点为（data，link），top是栈顶指针，若想删除栈顶结点，并将删除结点的值保存到x中，则应该执行的操作是（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | x=top->data; top=top->link; | B. | top=top->link; x=top->data; | | C. | x=top; top=top->link; | D. | x=top->link; | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 25. | 用链式方式存储的队列，在进行删除运算时（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 仅修改头指针 | B. | 仅修改尾指针 | | C. | 修改头指针和尾指针一定都要修改 | D. | 可能需要修改头指针和尾指针 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 26. | 循环队列存储在数组A[0,1,...,m]中，则入队时的操作是（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | rear=rear+1; | B. | rear=（rear+1）%（m-1）; | | C. | rear=（rear+1）%m; | D. | rear=（rear+1）%（m+1）; | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 27. | 栈和队列的共同点是（ ）。（1 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 都是先进先出 | B. | 都是先进后出 | | C. | 没有共同点 | D. | 只允许在端点处进行插入和删除操作 | |

**二. 多选题（共 8 分）**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. | 一个栈的输入序列为：a，b，c，d，e，则栈的可能输出序列是（ ）。（2 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | a,b,c,d,e | B. | d,e,c,b,a | | C. | d,c,e,a,b | D. | b,e,c,d,a | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2. | 设一个栈的入栈序列是1，2，3，4，5，则下列序列中，是合法的出栈序列的是（ ）。（2 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 5，1，2，3，4 | B. | 4，5，1，3，2 | | C. | 4，3，5，2，1 | D. | 2，3，5，4，1 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3. | 下列关于栈和队列的描述正确的是（ ）。（2 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 都是线性表 | B. | 都是操作受到限制的数据结构 | | C. | 数据元素之间具有一一对应的线性关系 | D. | 都是一种非线性的结构 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 4. | 下列关于栈和队列的描述不正确的是（ ）。（2 分）   |  |  |  |  | | --- | --- | --- | --- | | A. | 若入队顺序为 A,B,C,D，则出队顺序一定也是 A,B,C,D。 | B. | 若入栈顺序为 A,B,C,D，则出栈顺序一定也是 A,B,C,D。 | | C. | 以链表作为栈的存储结构，入栈操作必须判别栈满的情况。 | D. | 顺序栈入栈操作必须判别栈满的情况，出栈则无需判断栈空的情况。 | |

**三. 判断题（共 12 分）**

|  |  |
| --- | --- |
| 1. | 队列的特征是先进先出。（1 分） |

|  |  |
| --- | --- |
| 2. | 栈的特征是先进后出。（1 分） |

|  |  |
| --- | --- |
| 3. | 顺序队列会出现假溢出问题，解决的办法是用首尾相接的顺序存储结构，称为循环队列。（1 分） |

|  |  |
| --- | --- |
| 4. | 循环队列中，凡是涉及队首或队尾下标的修改都需要将其求模。（1 分） |

|  |  |
| --- | --- |
| 5. | 栈的链接存储结构称为链栈，通常用单链表表示，并且不用附加头结点。（1 分） |

|  |  |
| --- | --- |
| 6. | 若入队顺序为 A,B,C,D，则出队顺序一定也是 A,B,C,D。（1 分） |

|  |  |
| --- | --- |
| 7. | 栈和队列都是操作受限的线性结构。（1 分） |

|  |  |
| --- | --- |
| 8. | 以链表作为栈的存储结构，出栈操作必须判别栈空的情况。（1 分） |

|  |  |
| --- | --- |
| 9. | 栈和队列的存储方式既可是顺序方式，也可是链接方式。（1 分） |

|  |  |
| --- | --- |
| 10. | 以链表作为栈的存储结构，入栈操作必须判别栈满的情况。（1 分） |

|  |  |
| --- | --- |
| 11. | 在循环队列种，front指向队头元素的前一个位置，rear指向队尾元素位置，则队满的条件是front=rear。（1 分） |

|  |  |
| --- | --- |
| 12. | 有n个元素依次进栈，则出栈的序列有(n-1)/2种。（0.5 分） |

|  |  |
| --- | --- |
| 13. | 栈可以作为实现过程调用的一种数据结构。（0.5 分） |

**四. 填空题（共 28 分）**

|  |  |
| --- | --- |
| 1. | 设有一个空栈，栈顶指针为 1008，每个元素需要 2 个存储单元，则执行 push, push, pop, push, pop, push, push 后，栈顶指针为【1】。（1 分） |

|  |  |
| --- | --- |
| 2. | 若用于保存循环队列数据元素的数组最大长度为 8，front 为队首元素的前一个位置，当前值为 4，rear为队尾元素的位置，当前值为 2，则队列中的元素个数为【1】。（1 分） |

|  |  |
| --- | --- |
| 3. | 以下代码是顺序栈的初始操作和出栈操作，请填充完善。注意，不要填写多余的分号 （4 分） const int STACKSIZE = 10; template<class T> class SeqStack { public:  SeqStack(){top=【1】;}   T Pop(); //出栈操作函数   ....  private:  T data[STACKSIZE];   int top;  }; template <class T> T SeqStack<T>::Pop() {  if (top【2】) throw "栈下溢"; //空栈   top = 【3】; //修改栈顶元素的下标   return data[【4】]; //返回原栈顶元素；  } |

|  |  |
| --- | --- |
| 4. | \_\_\_\_\_\_\_结构可作为实现递归函数调用的一种数据结构，\_\_\_\_\_\_结构可作为打印机的缓冲区。请在以下答案中选择填空： （2 分）   线性表 集合 数组 顺序表 链表 栈 队列 |

|  |  |
| --- | --- |
| 5. | 已知链栈的结点结构如下： 栈顶指针为top，则实现将指针p所指结点插入栈顶的语句依次为【1】和【2】。 （2 分）  struct Node {  T data; //数据域   Node\* next; //指针域  }; |

|  |  |
| --- | --- |
| 6. | 对于栈和队列，无论它们采用顺序存储结构还是链接存储结构，进行插入和删除操作的时间复杂度都是【1】（1 分） |

|  |  |
| --- | --- |
| 7. | 在存储容量为 n 的循环队列中，为保证能正确地判断队列空或队列满，浪费一个数组单元，，队满时具有（ ）个元素。（1 分） |

|  |  |
| --- | --- |
| 8. | 设循环队列的容量为70，现经过一系列的入队和出队操作后，队头指针front为20，队尾指针rear为11，则队列中元素的个数为（ ）。（1 分） |

|  |  |
| --- | --- |
| 9. | 循环队列的引用目的是为了克服（ ）。 （1 分）  注：最多三个字 |

|  |  |
| --- | --- |
| 10. | 有一循环队列，存储队列的数组长度为50，队头指针为front，队尾指针为rear，判断满的条件是【1】，判断队空的条件是【2】。 （2 分） （注：在巩半角状态下输入，字符间不要加多余空格） |

|  |  |
| --- | --- |
| 11. | 若一个栈以向量V[0,1,...,n]存储，初始栈顶指针top设置为n+1，则元素x进栈时，top指针的正确操作是【1】（填写C++或C代码）。（1 分） |

|  |  |
| --- | --- |
| 12. | 若一个栈以向量V[0,1,...,n]存储，初始栈顶指针设置top=-1，则元素x进栈时，top指针的正确操作是【1】（填写C++或C代码）。（1 分） |

|  |  |
| --- | --- |
| 13. | 最大容量为n的循环队列，队尾指针是rear，rear指向当前队尾元素位置，队头指针是front，front指向当前队头元素的前一个位置，则元素x入队时，队尾指针rear的正确操作是【1】（填写C++或C代码）。（1 分） |

|  |  |
| --- | --- |
| 14. | 最大容量为n的循环队列，队尾指针是rear，rear指向当前队尾元素位置，队头指针是front，front指向当前队头元素的前一个位置，则元素x出队时，队头指针front的正确操作是【1】（填写C++或C代码）。（1 分） |

|  |  |
| --- | --- |
| 15. | 队列的特征是【1】。（1 分） |

|  |  |
| --- | --- |
| 16. | 栈的特征是【1】（请用四个字描述）。（1 分） |

|  |  |
| --- | --- |
| 17. | 设有一个空栈，栈顶指针为1000H，每个元素需要1个单位的存储空间，则执行push，push，pop，push，pop，push，push后，栈顶指针为【1】。（1 分） |

|  |  |
| --- | --- |
| 18. | 以下程序完成链栈的删除操作，栈顶指针top，请将程序补充完整： （3 分） T LinkStack<T>::Pop() {  Node<T> \*p=nullptr;   T x;   if (【1】） throw "下溢";   x=top->data;   【2】;   【3】;   delete p;   return x;  } |

|  |  |
| --- | --- |
| 19. | 以下程序完成链栈的插入操作，栈顶指针top，请将程序补充完整： （2 分） void LinkStack<T>::Push(T x) {  Node<T> \*s=nullptr;   s=new Node<T>;   【1】;   s->next=top;   【2】;  } |

**习题答案**

**一. 单选题（共 27 分）**

1.C 2.D 3.D 4.C 5.A 6.A 7.A 8.D 9.C 10.D 11.A 12.C 13.B 14.A 15.B  
16.B 17.A 18.D 19.B 20.B 21.C 22.C 23.A 24.A 25.D 26.D 27.D

**二. 多选题（共 8 分）**

1.AB 2.CD 3.ABC 4.BCD

**三. 判断题（共 12 分）**

1.√ 2.√ 3.√ 4.√ 5.√ 6.√ 7.√ 8.√ 9.√ 10.√ 11.× 12.×  
13.√

**四. 填空题（共 28 分）**

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | |  |  | | --- | --- | | 【1】 | 1014 | |

|  |  |  |  |
| --- | --- | --- | --- |
| 2. | |  |  | | --- | --- | | 【1】 | 6 | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3. | |  |  | | --- | --- | | 【1】 | -1 |  |  |  |  |  | | --- | --- | --- | --- | | 【2】 | ==-1 | <0 | <=-1 |  |  |  |  | | --- | --- | --- | | 【3】 | top-1 | --top |  |  |  |  | | --- | --- | --- | | 【4】 | top+1 | ++top | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 4. | |  |  | | --- | --- | | 【1】 | 栈 |  |  |  | | --- | --- | | 【2】 | 队列 | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 5. | |  |  |  | | --- | --- | --- | | 【1】 | p->next=top; | p->next=top |  |  |  |  | | --- | --- | --- | | 【2】 | top=p; | top=p | |

|  |  |  |  |
| --- | --- | --- | --- |
| 6. | |  |  | | --- | --- | | 【1】 | O(1) | |

|  |  |  |  |
| --- | --- | --- | --- |
| 7. | |  |  | | --- | --- | | 【1】 | n-1 | |

|  |  |  |  |
| --- | --- | --- | --- |
| 8. | |  |  | | --- | --- | | 【1】 | 61 | |

|  |  |  |  |
| --- | --- | --- | --- |
| 9. | |  |  | | --- | --- | | 【1】 | 假溢出 | |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10. | |  |  |  |  |  | | --- | --- | --- | --- | --- | | 【1】 | (rear+1)%50==front; | front==（rear+1)%50; | (rear+1)%50==front | front==（rear+1)%50 |  |  |  |  |  |  | | --- | --- | --- | --- | --- | | 【2】 | front==rear; | front==rear | rear==front; | rear==front | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 11. | |  |  |  |  |  | | --- | --- | --- | --- | --- | | 【1】 | top- - | top- -; | top=top-1; | top=top-1 | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 12. | |  |  |  |  |  | | --- | --- | --- | --- | --- | | 【1】 | top++ | top++; | top=top+1; | top=top+1 | |

|  |  |  |  |
| --- | --- | --- | --- |
| 13. | |  |  | | --- | --- | | 【1】 | rear=（rear+1）%n | |

|  |  |  |  |
| --- | --- | --- | --- |
| 14. | |  |  | | --- | --- | | 【1】 | front=（front+1）%n | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 15. | |  |  |  | | --- | --- | --- | | 【1】 | 先进先出 | 后进后出 | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 16. | |  |  |  | | --- | --- | --- | | 【1】 | 先进后出 | 后进先出 | |

|  |  |  |  |
| --- | --- | --- | --- |
| 17. | |  |  | | --- | --- | | 【1】 | 1003H | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 18. | |  |  |  | | --- | --- | --- | | 【1】 | top==nullptr | top==NULL |  |  |  | | --- | --- | | 【2】 | p=top |  |  |  | | --- | --- | | 【3】 | top=top->next | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 19. | |  |  | | --- | --- | | 【1】 | s->data=x |  |  |  | | --- | --- | | 【2】 | top=s | |