课时一：练习题

1. **数据结构形式的定义为(K, R)，其中K是（）的有限集合，R是K上关系的有限集合。**
   Answer: B. 数据元素
   Explanation: 数据结构的定义通常涉及元素和元素之间的关系，其中K是数据元素集合。

2. **数据元素是数据的最小单位。 ()**
   Answer: B. 数据元
   Explanation: 数据元素是数据的基本单位。

3. **存储数据时，通常不只保存数据元素的值，而还要保存（）。**
   Answer: D. 数据的存储方法
   Explanation: 除了保存数据的值，通常还需要保存数据的存储方式。

4. **排序树可以将数据结构分为（）。**
   Answer: C. 顺序结构和链式结构
   Explanation: 排序树通常分为顺序结构和链式结构。

5. **数据元素的逻辑关系来看，数据结构可分为四种：线性、集合、树和图，其中树形结构中的数据元素之间存在"_____"的关系。
   Answer: B. 层次关系
   Explanation: 树形结构中的数据元素有父子关系，形成层次结构。

6. **有图是一种非线性结构。 ()**
   Answer: A. 对
   Explanation: 图是一种非线性数据结构。

7. **以下属于顺序结构的是（）。**
   Answer: A. 顺序表
   Explanation: 顺序表是典型的顺序结构。

8. **以下是4个算法的时间复杂度函数表达式，其中时间复杂度最小的是（）。**
   Answer: D. $T(n) = n^2 + 1000$
   Explanation: 时间复杂度最小的是 $T(n) = n^2 + 1000$。

9. **算法是对特定问题求解步骤的描述，它具有（）特性，具有（）特性，输入和输出是重要的特性。
   Answer: 可行性、有效性
   Explanation: 算法的特性包括可行性和有效性。

10. **求下列程序段的时间复杂度。
    (1) for(i=0;i<n;i++)
    for(j=0;j<n;j++)
    A[i][j]=i+j;
    Answer: $O(n^2)$
    Explanation: 该程序有两个嵌套循环，时间复杂度为 $O(n^2)$。

(2) 代码段：

```c
y = 0;
while ((y + 1) * (y + 1) <= n) {
    y = y + 1;
}
```

- 循环条件是：`(y + 1) * (y + 1) <= n`，简化后就是 `y^2 <= n`，即 `y` 的平方小于等于 `n`。
- 这个循环将一直执行，直到 `y` 达到大约 `√n`，因为当 `y` 超过 `√n` 时，循环条件变为假。
- 因此，这个循环的时间复杂度是 **O(√n)**。

(3) 代码段：

```c
i = 1;
while (i <= n) {
    i = i * 3;
}
```

- 循环条件是 `i <= n`，并且每次迭代时，`i` 会乘以 3。
- 循环会一直执行，直到 `i` 超过 `n`。每次迭代时，`i` 以 3 为倍数增加，所以循环的次数大约是 $\log_3(n)$ 次，近似为 **O(log n)**（因为对数的底数对大 O 复杂度没有影响）。
- 因此，这个循环的时间复杂度是 **O(log n)**。

课时二：练习题

1. 线性表的特点是每个元素都有一个前驱和一个后继。

答案： B. 顺序存储
过程：
线性表通常是顺序存储结构的表现形式。在顺序存储中，每个元素都有前驱和后继元素。

2. 线性表的顺序存储结构是一种（）的存储结构。**

答案： B. 顺序存储
过程：
线性表的顺序存储结构是指数据元素在内存中按顺序排列，因此它是一种顺序存储结构。

3. 下列（）是顺序存储结构的优点。**

答案： A. 存储密度大
过程：
顺序存储结构的优点在于没有额外的指针开销，存储密度大，因为数据元素直接存储在连续的内存位置。

4. 两个有序顺序表分别有m个元素与n个元素，现将其归并成一个有序表，其最少的比较次数是（）。**

答案： C. n - 1
过程：
在归并两个有序顺序表时，最少的比较次数是 `n - 1`，因为每次归并时，至少有一个元素被选出。

5. 删除顺序表中第1个数据元素a0的时间复杂度是O(n)。

答案： A. 对
过程：
删除顺序表中的第一个元素后，需要将其后的所有元素向前移动一位，时间复杂度是O(n)。

6. 在n个元素的顺序表中，算法的时间复杂度是O(1)的操作是（）。

答案： C. 删除第i个元素（$1 \leq i \leq n$）

过程：

在顺序表中，删除指定位置的元素只需改变指针或标记，无需移动其他元素，因此时间复杂度是O(1)。

7. 下列（）不是线性表的特性。

答案： D. 线性表的长度等于n，并且n不等于0

过程：

线性表的长度不一定为n，它可以为空，长度为0也是有效的。

8. 二维数组A[10][20]按行优先存储，每个元素占4个存储单元，A[1][1]的存储地址是1000，A[5][6]的存储地址是_____。

答案： 1044

过程：

在按行优先存储的情况下，数组的存储地址可以通过以下公式计算：

$$\text{地址} = \text{基址} + (i \times \text{列数} + j) \times \text{单个元素占用空间}$$

这里，基址是1000，元素占用4个单元，列数是20。根据题目给出的索引 `A[5][6]`，计算过程如下：

$$\text{地址} = 1000 + (5 \times 20 + 6 - 1 \times 20 + 1) \times 4 = 1000 + 44 = 1044$$

9. 已知长度为n的线性表采用顺序存储结构。写一算法，删除线性表中所有值为x的元素。

答案：

过程：

可以通过遍历顺序表，将所有值为x的元素删除。每次删除时，其他元素向前移动一位。伪代码如下：

```c
for (i = 0; i < n; i++) {
    if (arr[i] == x) {
        for (j = i; j < n - 1; j++) {
            arr[j] = arr[j + 1];
        }
        n--;  // 更新线性表的大小
        i--;  // 保持当前索引
    }
}
```

时间复杂度：O(n)，因为我们遍历了整个数组并且在删除时移动了元素。

10. 设计一个高效算法，将顺序表中的所有元素逆置。

答案：

过程：

使用双指针法可以高效地逆置顺序表中的元素。伪代码如下：

```c
void reverseArray(int arr[], int n) {
    int start = 0, end = n - 1;
    while (start < end) {
        int temp = arr[start];
        arr[start] = arr[end];
        arr[end] = temp;
        start++;
        end--;
    }
}
```

时间复杂度是O(n)，因为只需要遍历数组一次。

课时三：练习题

1. 线性表采用链式存储时，其地址是（）。

答案： D. 连续与否都可以

过程：

在链式存储结构中，数据元素的存储地址不是连续的，链表是通过指针来连接不同的存储位置。因此，数据元素的存储地址可以是非连续的。

2. 顺序存储方式插入和删除元素效率太低，因此它不如链式存储方式好。（）

答案： B. 插入运算方便

过程：

顺序存储结构中，插入和删除元素时需要移动大量元素，效率低。而链式存储结构在插入和删除元素时只需要修改指针，因此效率更高。

3. **在双向链表中，若要求在p指针所指的结点之前插入指针所指的结点，则执行下列语句：s->next = p; s->prior = _____。

答案： D. p->prior = s;

过程：

双向链表中，插入操作需要调整指针。首先，`s->next = p`，然后设置`s->prior`为`p->prior`，并修改`p->prior = s`和`p->next = s`，以确保链表的结构正确。

4. 链表不具有的特点是（）。

答案： A. 插入不需要移动元素

过程：

链表具有插入和删除操作的优点，因为这些操作只需要改变指针，无需移动数据元素。而在数组等结构中，插入和删除通常需要移动元素。

5. 在一个单链表中，已知q结点是p结点的前驱结点，若在q和p之间插入结点s，则执行（）。

答案： C. s->next = p; s->prior = q;

过程：

插入结点时，需要修改指针。首先，设置`s->next = p`，然后设置`p->prior = s`，再将`q->next = s`，以确保链表结构正确。

6. 线性表的每个结点只能是一个单链表类型，而链表的每个结点可以是一个复合类型。（）。

答案： B. 用头指针表示的单循环链表

过程：

单链表的结点是单链结构，而循环链表的结点通过头指针连接，可以形成循环。

7. **将长度为n的单链表A与长度为m的单链表B之后的算法时间复杂度为_____。

答案： B. O(n + m)

过程：

将两个链表连接起来的时间复杂度是O(n + m)，因为需要遍历链表A的所有结点，并将链表B连接到A的末尾。

8. 对于在表的首尾两端进行操作的线性表，宜采用的存储结构是（）。

答案： D. 单链表

过程：

单链表适用于在两端进行操作，因为可以通过指针快速访问链表的首尾，不需要像数组那样频繁移动元素。

9. 链表的删除算法很简单，因为当删除链表中的某个结点后，计算机会自动将后续的各个单元前向移动。（）。

答案： A. 对

过程：

删除链表中的结点时，只需要调整指针，无需移动实际数据。因此，链表的删除操作通常比数组中删除操作简单。

10. 阅读下列算法，并补充所缺语句。

题目描述： 从头指针为la的带头结点的有序链表中删除所有值相同的多余元素，并释放被删除结点的空间。

c 复制

```c
void purge_linklst(ListLink *la){
    ListNode *p,*q,*t;
    ElemType temp;
    p=la->link;
    while(p!=NULL){
        q=p;
        temp=p->data;
        p=p->link;
        if(p!=NULL&&p->data==temp)
            p=p->link;
        else{
            while(p!=NULL&&p->data==temp){
                t=p;
                p=p->link;
                free(t);
            }
            p=q->link;
        }
    }
}
```

详细解答：

1. **解释算法过程：** 该算法用于从有序链表中删除所有重复的元素。算法使用两个指针：`p`用于遍历链表，`q`用于指向p的前一个节点。
   - 变量`temp`用于保存当前节点的值。
   - 第一个`while`循环遍历链表，判断当前节点的值是否与下一个节点的值相同。
   - 如果相同，跳过重复的节点并释放其内存；如果不同，更新指针，继续判断下一个节点。
2. **补充空缺的语句：**
   - `if(p!=NULL && p->data == temp)`
     这部分判断当前节点是否不为空且当前节点的值等于前一个节点的值，即判断是否是重复元素。如果是重复元素，则跳过该节点并继续检查下一个节点。
   - `while(p!=NULL && p->data == temp)`
     这部分在发现重复元素时，循环删除所有相同的节点，直到遇到值不同的节点为止。
3. **总结：** 该算法通过遍历链表并在找到重复元素时删除节点，确保链表中每个值都唯一。算法的时间复杂度是O(n)，其中n是链表中节点的数量。