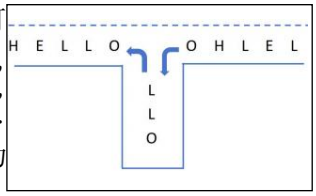
答案：1-5CBABC　6-10DABCB



# 高三上信息选择题提高专项练习（作业 40）

1. 某单词记忆游戏规则如下：玩家得到一个乱序的单词（如 OHLEL），可借助栈操作将其恢复为正确顺序（HELLO）（如图）。约定入栈标记为“I”，出栈标记为“O”，通过 IIIOOO 操作（如第 1 题图所示，字母‘H’、‘E’不入栈，直接进入最终队列）可恢复正确顺序。若要将“yphtno”恢复为正确顺序“python”，通过以下操作过程不能实现的是 C



第1题图

A. IOIOIO　　B. IIOOIOIO　　C. IOIIOOIOIO　　D. IIOOIIOOIIOO

2. 栈 S 初始为空，队列 Q 存在若干元素，将队列内元素依次出队，当栈空或出队元素与栈顶元素不相同时，出队元素入栈，否则出队元素重新入队。若队列最终为空，则初始队列 Q 从队首到队尾不可能为 B

A. abaab　　B. aabca　　C. accbb　　D. cabac

3. 某二叉树前序遍历结果为 ABCDEF，已知根节点的左右子树均为完全二叉树，则该二叉树后序遍历结果不可能是 A

A. CBDEFA　　B. CBEFDA　　C. BEDFCA　　D. DCEBFA

4. 某完全二叉树具有 2 个度为 2 的节点，且左子树的深度比右子树大 1。若将其补充为满二叉树后的中序遍历为 FEGDCBA，原完全二叉树的后序遍历为 B

A. GFEBD　　B. FGEBD　　C. CBGFED　　D. FGECBD

5. 定义如下函数：

```
def f(x):
    if len(x)==1:
        return ""
    if x[0]==x[-1]:
        return f(x[:-1])+x[0]
    return f(x[1:])
```

执行语句 t=f("abababb")后，t 的值为 C

A. "bbb"　　B. "bba"　　C. "abb"　　D. "aa"

1.x="abababb"：x[0]='a'≠x[-1]='b'→调用 f ("bababb")
2.x="bababb"：x [0]='b', x [-1]='b'→相同，调用 f ("babab")+"b"
3.x="babab"：x [0]='b', x [-1]='b'→相同，调用 f ("baba")+"b"
4.x="baba"：x [0]='b'≠x [-1]='a'→调用 f ("aba")
5.x="aba"：x [0]='a',x [-1]='a'→相同，调用 f ("ab")+"a"
6.x="ab"：x [0]='a'≠x [-1]='b'→调用 f ("b")。
7.x="b"：返回 ""
回溯拼接："abb"

6. 以下函数的功能是对列表 a 中元素升序排序

```
def sort(a):
    n=len(a)
    for i in range(1,n):
        key=a[i]
        j=i-1
        while _____:
            a[j+1]=a[j]
            j-=1
        a[j+1]=key
    return a
```

待插入值

划线处应填入的正确代码为： D

A. j>=0 and a[j]>a[j+1]　　B. j>=0 or a[j]<key　　C. j>0 or a[j]>a[j+1]　　D. j>=0 and a[j]>key

7. 有如下 python 程序段：

```
a=[7,3,2,4,6,9]
n=len(a)
i=n-1
while i>=1:
    k=0
```

```
for j in range(i):
    if a[j]<a[j+1]:
        a[j],a[j+1]=a[j+1],a[j]
        k=j
i=k
```

从前往后进行冒泡

0 ~ j 无序　　j+1 ~ n-1 有序

冒泡排序优化

若运行程序后，列表 a=[9,7,6,4,3,2]，则划线处的语句可以为①i-=1 ②i=k-1 ③i=k ④i=k+1　C

A.①③　　B.②③　　C.①④　　D.②④

8.某二分查找算法的 Python 程序段如下：

```
c=0; i,j=0,len(a)-1
while i<=j:
    m=(i+j)//2
    if key<=a[m]:
        j=m-1
    else:
        i=m+1
    c+=1
```

相等值左边界极限　该值比key值大

相等时往前找

key<a[m]相等时往后找

相等时往后找的停在此处

若非降序数组 a 包含 8 个元素，key 为 a 中一个元素，运行该程序段后 c 的值为 3。若将加框处代码修改为 key<a[m]，重新运行该程序段，c 的值依然为 3，则数组 a 中值为 key 的元素个数最多为　B

A.5　　B.6　　C.7　　D.8

9.有 Python 代码段如下：

```
k,h,t=10,0,0
n=len(nums); ans=n+1
a=[0]*(n+1); q=[0]*(n+1)
for i in range(n):
    a[i+1]=a[i]+nums[i]
for i in range(n+1):
    while h!=t and a[i]-a[q[h]]>=k:
        m=q[h]
        h+=1
        ans= min(ans,i-m)
    q[t]=i
    t+=1
print(ans)
```

功能：求最小长度的子数组和≥k

a数组存储前缀和

前缀和差值>=k,h+=1,出队

i-m是前缀和刚达到>=k标准的子序列长度

执行以上代码段，输出结果为 3，则 nums 可能为

A.[1,2,3,3,3,2,1]　　B.[2,2,2,2,2,2,2]　　C.[3,1,4,4,5,0,8]　　D.[9,7,6,3,5,5,1]

4　5　3　2

10.有如下 Python 程序段：

```
a=[2,-5,3,4,-1,2,3,5];qa=[0,0,0,0];max=0;head=1; tail=0;temp=0
for i in range(len(a)):
    qa[tail]=a[i]+temp-qa[head]
    if qa[tail]>max:
        max=qa[tail]
    temp=qa[tail]
    tail=(tail+1)%4
    head=(head+1)%4
print(max)
```

循环队列（最大子数组和）

```
初始化：qa=[0,0,0,0]，max=0，head=1，tail=0，temp=0。
循环 i=0 到 7：
i=0：qa[0]=2+0-qa[1]=2-0=2→max=2，temp=2，tail=1，head=2。
i=1：qa [1]=-5+2-qa [2]=-3-0=-3→max 保持 2，temp=-3，tail=2，head=3。
i=2：qa [2]=3+(-3)-qa [3]=0-0=0→max 保持 2，temp=0，tail=3，head=0。
i=3：qa [3]=4+0-qa [0]=4-2=2→max 保持 2，temp=2，tail=0，head=1。
i=4：qa[0]=-1+2-qa[1]=1-(-3)=4→max=4，temp=4，tail=1，head=2。
i=5：qa[1]=2+4-qa[2]=6-0=6→max=6，temp=6，tail=2，head=3。
i=6：qa[2]=3+6-qa[3]=9-2=7→max=7，temp=7，tail=3，head=0。
i=7：qa[3]=5+7-qa[0]=12-4=8→max=8，temp=8，tail=0，head=1。
最终 max=8。
```

运行程序，输出结果是　B

A.4　　B.8　　C.9　　D.10

答案1-5BCDCB 6-8ABC



# 高三上信息排序专题晚练

1.数组元素 a[0]~a[n-1] 已按升序排列，现要将 a[pos](0≤pos≤n-1)的值加 1，并保持数组的有序性不变，实现该功能的程序段如下，方框中应填入的正确代码为 B

t插入到pos~n-1区域之间（向后插入）

```
t = a[pos] + 1;i = pos
while [          ] :
    a[i] = a[i + 1]
    i = i + 1
a[i] = t
```

A. i<n-1 B. i<n-1 and t>a[i+1] C. i<n-1 and a[i]>a[i+1] D. i<=n-1 or t>a[i]

2. 有如下 python 程序段

D 1 2 3 4 5

```
d = [9, 3, 1, 8, 4, 2];s = [0, 1, 2, 3, 4, 5];n = len(d)
for i in range(1, n):
    key = s[i]
    j = i - 1
    while j >= 0 and d[s[j]] > d[key]:
        s[j + 1] = s[j]
        j -= 1
    s[j + 1] = key
    print(s)
```

插入排序的索引排序

前5个排序后：
1,3,4，8,9
对应下标：2，1,4,3,0

通过比较值，调整索引数组的元素位置

插入4次，一共可以排好5个数，只剩余d[5]未能参加排序，所以s[5]=5

执行该程序过程中，第 4 次（从 1 开始计数）输出的内容为C

A. [2,5,1,4,3,0] B. [2,1,3,0,4,5] C. [2,1,4,3,0,5] D. [2,1,0,3,4,5]

3. 有如下 Python 程序段：

```
a = [9, 7, 2, 5, 2, 8]
for i in range(len(a) - 1):
    for j in range(len(a) - 1, i, -1):
        if a[j] > a[j - 1]:  # 语句①
            a[j], a[j - 1] = a[j - 1], a[j]
```

执行上述程序段后，下列说法正确的是 D

A. 语句①的执行次数为 30 比较次数n*(n-1)/2=15

B. 若问题规模为 n，则该程序段的时间复杂度为 $O(\log_2 n)$ $O(n^2)$

C. 将代码中的“-1”改为“+1”，可以实现数组 a 升序排序 出现a[len(a)],下标越界

D. 将加框处的语句修改为“range(1, len(a) - i)”，程序执行的结果无变化

4. 有如下 Python 程序段：

[[x,x,x,x],
[x,x,x,x],
[x,x,x,x],]

```
b = []; n = len(a)
for i in range (n):
    for j in range (len (a [i])):
        if i < j and a [i][j] not in b:
            b.append (a [i][j])
```

执行该程序段后，若数组 b 的值为 [3,6,7,9]，那么数组 a 的值不可能的是 C

A. [[1,3,6,7],[2,4,7,9],[5,2,3,7]] B. [[1,3,6,7],[1,3,6,7],[1,3,6,9]]

C. [[1,3,6,7],[2,4,9,7],[5,2,3,8]] D. [[1,3,6,7],[2,6,9,7],[5,2,3,3]]

5. 有如下 Python 程序段：

```
d=[12,45,24,63,2,3,60,50]
step=int (input ("输入 1~4 之间的整数:"))
for i in range (step,len (d)):
    j=i-step
```

```
    t=d [i]
    while j >=0 and t>d [j]:
        d [j+step]=d [j]
        j=j-step
    d[j+step]=t
print (d)
```

希尔排序（分组插入排序），步长 step=[1,4]

通过插入排序的方式，以 step 为间隔进行降序排序。
当 step=1 时，索引相差 1 之间的元素进行排序，得到[63, 60, 50, 45, 24, 12, 3, 2]，排除 A 项；
当 step=2 时，索引相差 2 之间的元素进行排序，得到[60, 63, 24, 50, 12, 45, 2, 3]，排除 D 项；
当 step=3 时，索引相差 3 之间的元素进行排序，得到[63, 50, 24, 60, 45, 3, 12, 2]；
当 step=4 时，索引相差 4 之间的元素进行排序，得到[12, 45, 60, 63, 2, 3, 24, 50]，排除 C 项。

运行该程序段，输出结果不可能是 B

A.[63, 60, 50, 45, 24, 12, 3, 2] step=1　B. [63, 50, 24, 60, 45, 2, 12, 3] step=3

C.[12, 45, 60, 63, 2, 3, 24, 50] step=4　D. [60, 63, 24, 50, 12, 45, 2, 3] step=2

6. 某 Python 程序段如下所示：

```
import random
lst = [2, 1, 7, 5, 5, 9, 8, 9, 3, 8];b = [0] * 10
for i in range (10):
    b [lst [i]] += 1
j=0
k = random.randint (3, 6)
while k > b [j]:
    k -= b [j]; j += 1
```

投桶　b数组

k=6，j=7
k=4，j=5
k=3，j=3

| 数值 | 1 | 1 | 1 | 0 | 2 | 0 | 1 | 2 | 2 |
|---|---|---|---|---|---|---|---|---|---|
| 索引 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

b数组中值累加达到刚达到或超出k的范围,就是j停止的地方

运行该程序，j 的值不可能为 A

A. 8　　B. 7　　C. 5　　D. 3

7. 有如下 Python 程序段：

```
#随机生成数组 d，代码略
for i in range (len (d)-1):
    for j in range (　　) :
        if d [j] > d [j+1]:
            d [j], d [j+1] = d [j+1], d [j]
```

冒泡排序内循环口诀：初定终变
看if定初值
代外循环终值等于内循环初值(x+/-i=内循环初值)

加框处可选取的代码：①0,len(d)-i-1　②1,len(d)-2　③len(d)-1,i,-1　④len(d)-2,i-1,-1

以下能正确实现排序的是 B

A. ①③　　B. ①④　　C. ②③　　D. ②④

8. 已知列表 a 中存储了 n 个整型数据，小孙设计了一个算法可以删除列表 a 中重复的数据，并将去重后的数据升序排序。实现该功能的程序段如下，方框中应填入的正确代码为 C.

```
bottom = len (a); i = 0
while i < bottom - 1:
    flag = False; j = bottom - 2
    while j >= i:
        if a [j+1] < a [j]:
            a [j], a [j+1] = a [j+1], a [j]
            flag = True
        elif a [j] == a [j+1]:
            (　　)
            flag = True
        j -= 1
    if not flag:
        break
    i += 1
```

发现重复元素（a[j]==a [j+1]）时，需将重复元素（a[j+1]）移到数组末尾（bottom-1位置），并减小bottom（排除已删除元素）

从后往前的冒泡排序，
a[j]和a[j+1]相等时，如果用bottom-1换掉a[j],此时a[j]是从后到j之间交换好的数，但a[bottom-1]不符合，所以不可以换

A. a [j], a [bottom-1] = a [bottom-1], a [j];bottom -= 1
B. a [j], a [bottom-1] = a [bottom-1], a [j];j += 1
C. a [j+1], a [bottom-1] = a [bottom-1], a [j+1];bottom -= 1
D. a [j+1], a [bottom-1] = a [bottom-1], a [j+1];j += 1