49(1)2 (2)①a[i]==0 ②a[i]==a[mx] ③cnt[mx]=0

36.有如下 Python 程序段：

```
for i in range(1, len(data)):
    if data[i] < data[i - 1]:
        data[i], data[i - 1] = data[i - 1], data[i]
```

所有数都参加，只比了一趟

运行该程序段后，列表 data 中的数据按升序排列，则运行前的 data 可能是 D

A. [1,3,4,2,5,6]　　B. [3,2,1,4,5,6]　　C. [1,5,3,4,2,6]　　D. [2,1,3,5,4,6]

37.数组元素 a [0]~a [n-1] 中数据已按升序排列，现将 a[p]（0≤p<n-1）的值减 2，数组 a 仍保持有序，实现该功能的程序如下，方框中应填入的正确代码为 B

```
t = a[p] - 2
i = p
while [          ]:
    a[i] = a[i - 1]:
    i -= 1
a[i] = t
```

插入排序，
只有一个p位上的数需要进行插排

A. i > 0　　B. i > 0 and t < a[i - 1]

C. i > 0 and a[i] < a[i - 1]　　D. i > 0 or t < a[i - 1]

38.有如下 Python 程序段：

```
import random
q = [0, 0, 0, 0, 0]
head = tail = 3
while head != (tail + 1) % 5:
    q[tail] = random.randint(1, 9)
    if q[tail] % 2 == 0:
        tail = (tail + 1) % 5
```

0 1 2 3 4

循环队列一共5个元素，tail占了一个空位，
head=3，只有偶数能入队，最后一个位置2一定是0

执行该程序段后，q 的值可能是 A

A. [4,4,0,4,4]　　B. [4,8,2,4,6]

C. [4,0,4,4,6]　　D. [0,0,0,4,6]

39.定义如下函数：

```
def f(x, y):
    if x >= y:
        return 1
    print("#")
    return x * f(x, y - 1) + y * f(x + 1, y)
```

#

f(1,3)=1*f(1,2)+3*f(2,3)

=1*(1*f(1,1)+2*f(2,2))+3*(2*f(2,2)+3*f(3,3))

# #

执行语句 v = f (1, 3) 后，输出 "#" 的个数为 C

A. 0　　B. 1　　C. 3　　D. 7

40.定义如下函数：

```
def f(a, n, m):
    if n == 1:
        return a
    elif n % m == 0:
        a.append(m)  # 为列表 a 追加一个元素 m
        return f(a, n // m, m)
    else:
        return f(a, n, m + 1)
```

对n进行质因数分解

列表 a 的初始值为[]，执行语句 f (a, 12, 2) 后，a 的值为 B

A. [2,4,6]　　B. [2,2,3]　　C. [2,3,6]　　D. [2,3,4,6]

41.有如下 Python 程序段：

```
def pal(s):
    if len(s) <= 1:
        return True
    elif s[0] == s[len(s) - 1]:
        return pal(s[1:len(s) - 1])
    else:
        return False
print(pal(s))
```

回文字符串判断

若 s 为 "TTQAQTT"，运行上述程序后，下列说法正确的是 B

A. 程序输出结果是 FalseTrue　B. 最后一次调用 pal 时参数 s 的值是 "A"

C. 函数 pal 被调用了 3 次4次　D. 画线处的代码若改为 "len(s)==0"，pal 函数被调用的次数不变 多一次

42.对于任意正整数 x，甲、乙两个程序段输出的结果都相同，则乙程序段方框中正确的代码为 B

甲程序段	乙程序段

def f(x):
　if x == 1:
　　return str(x)
　else:
　　return f(x // 2) + str(x % 2)
print(f(x))
	s = ""
	while x > 0:
	　[　　]
	print(s)

十进制转二进制
余数倒序读

A. ①s = s + str(x % 2)　②x = x // 2

B. ①s = str(x % 2) + s　②x = x // 2

C. ①x = x // 2　②s = str(x % 2) + s

D. ①x = x // 2　②s = s + str(x % 2)

43.定义如下函数：

```
def binSearch(data, key, i, j):
    while i <= j:
        m = (i + j) // 2
        if data[m] == key:
            return m
        elif data[m] < key:
            i = m + 1
        else:
            j = m - 1
    return -1
```

0 1 2 3 4 5 6 7 8 9

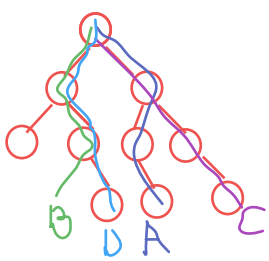
若 d 为 [6,12,15,18,22,25,35,46,58,60]，k 为 25，分别执行下列语句，函数中画线处语句的执行次数最少的是 A

A. p = binSearch (d, k, 0, 6) 2次　B. p = binSearch (d, k, 1, 7) 3次

C. p = binSearch (d, k, 1, 8) 3次　D. p = binSearch (d, k, 3, 9) 3次

44.有如下 Python 程序段：

```
# 随机产生10 个整型元素的非降序序列存入列表 a，代码略
i, j = 0, len(a) - 1 ; key = int(input()) ; b = []
while i <= j:
    m = (i + j) // 2
    b.append(a[m])
    if a[m] > key:
        j = m - 1
    else:
        i = m + 1
print(b)
```



运行该程序段后，b 的结果不可能是 B

A. [14,17,14,16]　B. [11,5,9,6]　C. [13,19,19,19]　D. [10,5,7,8]

```
    else:
        i = m + 1
print(b)
```

运行该程序段后，b 的结果不可能是 B

A. [14,17,14,16]　　B. [11,5,9,6]　　C. [13,19,19,19]　　D. [10,5,7,8]

45.有如下 Python 程序段：

```
from random import randint
i, j = 0, len(s)
st = []
while i < j:
    if randint(0, 1) == 0:
        s += s[i]
        j += 1
    else:
        if len(st) > 0 and s[i] < st[len(st) - 1]:
            st[len(st) - 1] = s[i]
        else:
            st.append(s[i])
    i += 1
```

A 0 1 0 0 1 1 1
a b c d a c d
覆盖

B 0 1 1 0 1 1
a b c d a d
覆盖

C 0 1 1 1 1
a b c d a
覆盖

若 s 为 "abcd"，运行该程序段后，列表 st 不可能是 D

A. ["a","c","d"]　　B. ["b","a","d"]　　C. ["b","c","a"]　　D. ["c","a","d"]

46.用如下 Python 程序段实现对数组元素 a [0] 到 a [9] 从小到大排序。

c不可能覆盖b,a不能跳过c覆盖b

```
i = 0
while i < 9:
    k, i = i, 10
    for j in range(9, k, -1):
        if __(1)__:
            a[j], a[j - 1] = a[j - 1], a[j]
            __(2)__ i=j
```

冒泡排序优化

0~j-1 有序　j~9 无序

上述程序段中画线处的可选代码有：

①a [j] < a [j - 1]　②a [j] > a [j - 1]　③i = j　④i = k - 1　⑤i = i + 1

则 (1)(2) 处代码序号依次为 A

A. ①③　　B. ①④　　C. ②④　　D. ②⑤

47.有如下 Python 程序段：

25年6月真题第11题同类型

```
import random
tag = [""] * len(data)
p = i = 0
while i < len(data):
    if tag[p] == "" and p != -1:
        tag[i] += data[p][0]
        p = data[p][1]
    else:
        tag[i] += tag[p]
        i += 1
        p = i
```

记录从当前节点开始到尾节点字符连接

0 1 2 3 4

若 data 为 [["A",3],["B",-1],["C",0],["D",1],["E",2]]，运行该程序段后，tag [4] 的值为 D

A. "ADB"　　B. "CADB"　　C. "DB"　　D. "ECADB"

tag=["ADB","B","CADB","DB","ECADB"]
0 1 2 3 4

48.使用列表模拟 dba 和 dbb 两个链表结构（节点数均大于 0），每个节点含数据域和指针域，两个链表的头指针分别为 ha 和 hb，链表中所有节点已按数据域降序排列，如第 48 题图所示。现要将 dbb 中数据域大于 dba 中数据域最小值的节点合并到 dba 中，并保持链表 dba 的有序性。实现该功能的程序段如下。

```
if dbb[hb][0] > dba[ha][0]:
    dba.append([dbb[hb][0], ha])
    ha = len(dba) - 1
    hb = dbb[hb][1]
q = ha
while hb != -1 and q != -1:
    p = dba[q][1]
    if p != -1 __(1)and__ dbb[hb][0] > dba[p][0]:
        dba.append([dbb[hb][0], p])
        dba[q][1] = len(dba) - 1
        hb = dbb[hb][1]
    __(2)q=dba[q][1]__
```

缩进调整

二路归并，将dbb中的节点插入到dba中

	数据区域	指针区域
0	58	-1
1	68	2
2	65	4
3	72	1
4	62	0
ha→ 5	73	3

	数据区域	指针区域
0	52	-1
1	64	6
2	65	1
3	58	5
hb→ 4	67	2
5	53	0
6	60	3

第 48 题图

如果dbb中的值比dba当前节点大，就插入在当前dba节点之前

p是dba链中的对比节点指针，hb是dbb链中的对比节点指针，q是dba完成和dbb对比后的链尾指针

（1）与（2）处可选的表达式有：①and　②or　③q = dba [q][1]　④q = p

则（1）与（2）处的表达式序号依次为

A. ①③　B. ①④　C. ②③　D. ②④

若dbb中的节点没有插入到dba中q=p可以成立
若dbb插入到dba中，q=p会让p节点失去与dbb中节点对比的机会

高三上信息技术：综合练习

49.根据机器的负载率对工厂的 n 台机器［编号 0~(n-1)］进行监控和调度。调度规则是：每隔 1 小时采集 1 次各台机器的负载率（负载率用百分制表示，例如，负载率 95% 表示为 95，机器休息时的负载率为 0），调度负载率最高的机器休息；若有多台机器的负载率同为最高，则调度连续工作时间最长的机器中编号最小的机器休息。休息的机器在休息 1 小时后再次工作。请回答下列问题：

(1) 若共有 10 台机器，某次采集到 0~9 号机器的负载率和连续工作的时长如第 49 题图所示，当前 7 号机器处于休息状态，则接下来需要调度休息的机器编号是 ____2____。

机器编号	0	1	2	3	4	5	6	7	8	9
负载率	93	85	93	93	82	85	93	0	82	93
连续工作时长/小时	5	3	6	1	4	7	2	0	2	6

第 49 题图

(2) 实现上述功能的部分 Python 程序如下，请在画线处填入合适的代码。

```
n = 10  # 共有 10 台机器
a = [0] * n  # 列表 a 的长度为 n，各元素值均为 0，a[i]用于存放 i 号机器的负载率
cnt = [0] * n  # cnt[i]用于存放 i 号机器的连续工作时长（单位为小时）
# 启动 1~(n-1)号机器工作，0 号机器休息，代码略
while True:
    # 延时 1 小时，采集各台机器的负载率存入 a，代码略
    mx = 0
    for i in range(n):
        if ____①a[i]==0____:
            # 调度 i 号机器工作，代码略
        elif a[i] > a[mx]:
            mx = i
        elif ____②a[i]==a[mx]____ and cnt[i] > cnt[mx]:
            mx = i
    for i in range(n):
        cnt[i] = cnt[i] + 1
    # 调度 mx 号机器休息，代码略
    ____③cnt[mx]=0____
```

如果当前机器在休息的，让它开始工作

负载率相同时，找工作时长长的，如果工作时长也相同，则看编号小的，i循环由小到大，mx本身记录的下标小于等于i,所以不用写工作时长相同时比较编号的条件

休息的机器连续工作时长被打断，从0开始记