1. **单选**

3．假设现在有一个整形变量（Number），它被限制为32位，那么它将占用（）个字节。

A. 32 B 128 C. 4 D. 8

答案：C

注释：可以不改题目，改了说法更准确点。

4．若有如下程序段， 其中s、 a、 c均为整型变量，且a, c均己赋值（c 大于0)

s = a

for b in range(c):

s = s – 1

则与上述程序段功能等价的赋值语句是

A. S=a-c; B. s=a-b; C. s=s-c; D. S=b-c

1. **阅读程序**

**1．**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | st = input() # str类型  n = len(st)  st = list(st)  i = 1  while i <= n:  if n % i == 0:  c = st[i-1]  if c >= 'a':  st[i-1] = chr(ord(c) - ord('a') + ord('A'))  i += 1  print(''.join(st)) |

**判断题：**

**2）若将第4行的i=1改为i=0，程序运行会发生错误()答案：对**

**3）若将第5行的i <= n改为i\*i <= n，程序运行会发生错误()答案：对**

**2．**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | n, m = list(map(int, input().split()))  a, b = [0]\*(n+1), [0]\*(n+1)  for i in range(1, m+1):  x, y = list(map(int, input().split()))  if a[x] < y and b[y] < x:  if a[x] > 0:  b[a[x]] = 0  if b[y] > 0:  a[b[y]] = 0  a[x] = y  b[y] = x  ans = 0  for i in range(1, n+1):  if a[i] == 0:  ans += 1  if b[i] == 0:  ans += 1  print(ans) |

n, m都是正整数，x, y都是[1, n]范围内的整数

**判断题：**

**2）若执行完第17行的ans += 1时，ans一定是偶数()答案：错**

**4）若程序执行到第5行，x总是小于y，那么第7行不会被执行()答案：错**

**3．**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | a, b = list(), list()  MAXN = 10000  def f(l: int, r: int, depth: int):  if l > r:  return 0  min\_, mink = MAXN, 0  for i in range(l, r+1):  if min\_ > a[i]:  min\_ = a[i]  mink = i  lres = f(l, mink-1, depth+1)  rres = f(mink+1, r, depth+1)  return lres+rres+depth\*b[mink]  n = int(input())  a = list(map(int, input().split()))  b = list(map(int, input().split()))  print(f(0, n-1, 1)) |

**选择题：**

**3）当n=100时，最坏的情况下，与第8行的比较运算执行从次数最接近的是：（）**

**A. 5000 B. 600 C. 6 D. 100**

**答案：A**

**4）当n=100时，最好的情况下，与第8行的比较运算执行从次数最接近的是：（）**

**A. 100 B. 6 C. 5000 D. 600**

**答案：D**

1. **完善程序**
2. **矩阵变幻**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | MAX\_SIZE = 1 << 10  n, res = 0, [[0]\* MAX\_SIZE for i in range(MAX\_SIZE)]  def recursive(x: int, y: int, n: int, t: int):  if n == 0:  res[x][y] = ①  return  step = 1 << (n-1)  recursive(②, n-1, t)  recursive(x, y+step, n-1, t)  recursive(x+step, y, n-1, t)  recursive(③, n-1, not t)  n = int(input())  recursive(0, 0, ④)  size = ⑤  for i in range(size):  for j in range(size):  print(int(res[i][j]), end='')  print("") |

1. **计数排序**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | maxn = 10000000  maxs = 10000  a, b, res, ord, cnt = [0]\*maxn, [0]\*maxn, [0]\*maxn, [0]\*maxn, [0]\*(maxs+1)  n = int(input())  for i in range(n):  a[i], b[i] = list(map(int, input().split()))  for i in range(n):  ① # 利用cnt数组统计数量  for i in range(maxs):  cnt[i+1] += cnt[i]  for i in range(n):  ② # 记录初步排序结果  cnt = [0]\*(maxs+1)  for i in range(n):  ③ # 利用cnt数组统计数量  for i in range(maxs):  cnt[i+1] += cnt[i]  for i in range(n-1, -1, -1):  ④ # 记录最终排序结果  for i in range(n):  print("%d %d" % (⑤)) |

**1）**①处应该填（）

**A. cnt[i] += 1**

**B. cnt[b[i]] += 1**

**C. cnt[a[i] \* maxs + b[i]] += 1**

**D. cnt[a[i]] += 1**

**2）**②处应该填（）

**A. cnt[a[i]] -= 1**

**ord[cnt[a[i]]] = i**

**B. cnt[b[i]] -= 1**

**ord[cnt[b[i]]] = a[i]**

**C. cnt[a[i]] -= 1**

**ord[cnt[a[i]]] = b[i]**

**D. cnt[b[i]] -= 1**

**ord[cnt[b[i]]] = i**

**3）**③处应该填（）

**A. cnt[b[i]] += 1**

**B. cnt[a[i] \* maxs + b[i]] += 1**

**C. cnt[a[i]] += 1**

**D. cnt[i] += 1**

**4）**④处应该填（）

**A. cnt[a[ord[i]]] -= 1**

**res[cnt[a[ord[i]]]] = ord[i]**

**B. cnt[b[ord[i]]] -= 1**

**res[cnt[b[ord[i]]]] = ord[i]**

**C. cnt[b[i]] -= 1**

**res[cnt[b[i]]] = ord[i]**

**D. cnt[a[i]] -= 1**

**res[cnt[a[i]]] = ord[i]**