1: C
解释:
前两个数比较，大的为最大值，小的为最小值，用掉 1 次；

还剩下 2*（n-1)个数，每两个比较，大的再和最大值比较，小的再和最小值比较，一共是 3*（n-1) 次

所以加一起就是（3*n-2）次

2: 102
3:1075
4: 42
5: 9
6: 4
解释:
32 位总线跨度可以寻址 2^32B,1GB=2^30B;所以 2^32B=2^2GB=4GB;
字长：在同一时间中处理二进制数的位数叫字长。通常称处理字长为 8 位数据的 CPU 叫 8 位 CPU，32 位 CPU 就是在同一时间内处理字长为 32 位的二进制数据。二进制的每一个 0 或 1 是组成二进制的最小单位,称为一个比特（bit）。

一般说来，计算机在同一时间内处理的一组二进制数称为一个计算机的"字"，而这组二进制数的位数就是"字长"。字长与计算机的功能和用途有很大的关系，是计算机的一个重要技术指标。字长直接反映了一台计算机的计算精度，为适应不同的要求及协调运算精度和硬件造价间的关系，大多数计算机均支持变字长运算，即机内可实现半字长、全字长（或单字长）和双倍字长运算。在其他指标相同时，字长越大计算机的处理数据的速度就越快。早期的微机字长一般是 8 位和 16 位，386 以及更高的处理器大多是 32 位。目前市面上的计算机的处理器大部分已达到 64 位。

字长由微处理器（CPU）对外数据通路的数据总线条数决定。

最小可寻址单位：内存的最小可寻址单位通常都是字节。也就是说一个指针地址值可对应内存中一个字节的空间。

寻址空间：寻址空间一般指的是 CPU 对于内存寻址的能力。CPU 最大能查找多大范围的地址叫做寻址能力，CPU 的寻址能力以字节为单位 （字节是最小可寻址单位），如 32 位寻址的 CPU 可以寻址 2 的 32 次方大小的地址也就是 4G，这也是为什么 32 位寻址的 CPU 最大能搭配 4G 内存的原因，再多的话 CPU 就找不到了。

这里 CPU 的寻址位数是由地址总线的位数决定，32 位 CPU 的寻址位数不一定是 32 位，因为 32 位 CPU 中 32 的意义为字长。

有关寻址范围计算解释，对于 32 位寻址的 CPU，其地址值为 32 位的二进制数，所以可以表示的最大地址为 2 的 32 次方(即 4G，最大内存空间为 4GB，这里 G 表示数量、GB 表示容量)。同时我们不难看出，一个指针的值就是一个 32 位的二进制数，32 位对应 4 字节(Byte)。所以，指针的大小实际上是由 CPU 的寻址位数决定，而不是字长。

7: 55
8: 3
从上表可以看出：

可将每门考试科目标号，并作为图的顶点画图，根据每位学生需要考试的科目，连接顶点与顶点，考三门连三条边，考两门则连一条边。形成该图的至少三个独立集：{1,3,5}{2,7}{4,6}，使所有考试安排不冲突。所以：

考第一门时，可以同时考第三门、第五门（或第七门）

考第二门时，可以同时考第七门（或第五门）

考第四门时，可以同时考第六门

即最少安排 3 个不同的考试时间段才能避免冲突。

9: D
解释
稳定的排序基数排序、冒泡排序、直接插入排序、折半插入排序、归并排序
不稳定：堆排序、快速排序、希尔排序、直接选择排序
10: 3
11: 4 9
12: 9
13: 256
三个变量，每个变量可取 0,1 两种值，共有 2^3=8 种组合；任意一个变量组合代入表达式，只有 0 和 1 两种值。因此两两不等价的表达式最多有 2^8=256 种。

14: 5536
解释:
图 1 的独立集：{∅ }{1}{2}{3}{2,3}
图 2 的独立集：{∅ }{1}{2}{3}{4}{5}{1,4}{1,5}{1,4,5}{2,3}{3,4}{3,5}{3,4,5}{4,5}
图 3 可使用 DP 求解：设 m(i)为以 i 号点为根结点的总个数；f(i)为选 i 的总个数；g(i)表示不选 i 的总个数，则有：m(i)=f(i)+g(i)
f(i)=g(left_child[i])*g(right_child[i])
g(i)=m(left_child[i])*m(right_child[i])
m(17)=f(17)+g(17)=1936+3600=5536
f(17)=g(8)*g(8)=44*44=1936
g(17)=m(8)*m(8)=60*60=3600
m(8)=f(8)+g(8)=16+44=60
f(8)=g(1)*g(6)=1*16=16
g(8)=m(1)*m(6)=2*22=44
m(6)=f(6)+g(6)=6+16=22
f(6)=g(1)*g(4)=1*6=6
g(6)=m(1)*m(4)=2*8=16
m(4)=f(4)+g(4)=2+6=8

f(4)=g(1)*g(2)=1*2=2
g(4)=m(1)*m(2)=2*3=6
m(2)=f(2)+g(2)=3
f(2)=g(1)=1
g(2)=m(1)=2
m(1)=2
f(1)=1
g(1)=1

15: 55

16: 454
根据常识 (a|b) + (a&b) = a+b
令 a|b = x, a&b = y
那么
x+y =a+b
x*y = a*b
联立解方程可得 y=a 或者 y=b
因为 x>=y 所以 x=max(a,b), y = min(a,b)
所以 min(a,b)一定是 max(a,b)的子集
得出这个结论后利用组合数算一算就好了
17: 17 24 1 8 15
18:
p[0]
rest<q 或 q>rest
rest/q
rest%q*10+p[i]
rest%q
19:
i<=j
next[rank[i]]=rank[i+1]
higher=height[next[i]]-height[i]
shorter<higher
previous[next[i]]=previous[i]
20: 31
21:
j-1
cur1

count1-- / count1 = count1 − 1 / --count1
count2-- / count2 = count2 − 1 / --count2
cur1 = a[j]
22:

n-p+i / i-p+n
i-p+1
a[i-p]
j<=end2
i / start2 / end1+1
j-1
23: 55
24:
false
used[data[i]]=false
j
n
Break
25:
return(k%c)+1
s[n]=q[tail]
q[head]
q[head]
q[tail]
next(head)