NOIP2018 初赛提高组真题及答案

说明：难题解析比较详细，简单题没有解析

一、单项选择题（共 10 题，每题 2 分，共计 20 分；每题有且仅有一个正确选项）

1. 下列四个不同进制的数中，与其它三项数值上不相等的是（ ）。
A. (269) 16
B. (617) 10
C. (1151) 8
D. (1001101011) 2

答案：D
解析：考察进制转换，我们可以先将 A,B 转换为二进制，就可以发现 A,B 相等，但是与 D 不同，所以选 D

2. 下列属于解释执行的程序设计语言是（ ）。
A. C
B. C++
C. Pascal
D. Python

答案：D
解析：因为 C,C++,pascal 都是需要编译的语言，非解释性语言
而 python 是交互式的，也是解释性语言

3. 中国计算机学会于（ ）年创办全国青少年计算机程序设计竞赛。
A. 1983
B. 1984
C. 1985
D. 1986

答案：B

4. 设根节点深度为 0，一棵深度为 h 的满 k（k>1）叉树，即除最后一层无任何子节点外，每一层上的所有结点都有 k 个子结点的树，共有（ ）个结点。
A. $(k^{h+1} - 1) / (k - 1)$
B. k^{h-1}
C. k^{h}
D. $(k^{h-1}) / (k - 1)$

答案：A
解析：等比数列求和，

5. 设某算法的时间复杂度函数的递推方程是 T(n) = T(n - 1) + n（n 为正整数）及 T(0) = 1，则该算法的时间复杂度为（ ）。
A. O(log n)
B. O(n log n)
C. O(n)
D. O(n2)

答案：D
解析：

$$T(n)-T(n-1)=n$$

，求和：

$$\sum_{i=1}^{n}(T(i)-T(i-1))=\sum_{i=1}^{n}i=\frac{n(n+1)}{2}$$

6. 表达式 a * d - b *c 的前缀形式是（ ）。
A. a d * b c * -
B. - * a d * b c
C. a * d - b * c
D. - * * a d b c
答案：B
解析：先建一棵表达式树，其先序遍历就是前缀表达式

7. 在一条长度为 1 的线段上随机取两个点，则以这两个点为端点的线段的期望长度是（ ）。
A. 1 / 2
B. 1 / 3
C. 2 / 3
D. 3 / 5

答案：B
解析：我们先考虑固定一个端点的情况，如果左端点固定在了最左边那么答案为 1/2，既然左端点更大，那么肯定答案会小于 1/2，因此只能是 B

熟悉 ODT（即珂朵莉树）理论的同学可以发现 ODT 复杂度证明里面有这个东西，即随机意义下期望区间长度。

8. 关于 Catalan 数 Cn = (2n)! / (n + 1)! / n! ，下列说法中错误的是（ ）。

A. Cn 表示有 n + 1 个结点的不同形态的二叉树的个数。
B. Cn 表示含 n 对括号的合法括号序列的个数。
C. Cn 表示长度为 n 的入栈序列对应的合法出栈序列个数。
D. Cn 表示通过连接顶点而将 n + 2 边的凸多边形分成三角形的方法个数。

答案：A
解析：对于 A，令 n=1，2 个节点的二叉树形态有 2 种，但是 C1=1，显然错误

9. 假设一台抽奖机中有红、蓝两色的球，任意时刻按下抽奖按钮，都会等概率获得红球或蓝球之一。有足够多的人每人都用这台抽奖机抽奖，假如他们的策略均为：抽中蓝球则继续抽球，抽中红球则停止。最后每个人都把自己获得的所有球放到一个大箱子里，最终大箱子里的红球与蓝球的比例接近于（ ）。
A. 1 : 2
B. 2 : 1
C. 1 : 3
D. 1 : 1

答案：D

解析：一个人在第 i 轮可以得到的红球期望数量为：$\frac{1}{2^i}$，而 $\sum_{i=1}^{\infty}\frac{1}{2^i}=1$ 所以每个人得到红球期望数量为1，而得到蓝球数量必定为1，所以为1:1

10. 为了统计一个非负整数的二进制形式中 1 的个数，代码如下：

```
int CountBit(int x)
{
  int ret = 0;
  while (x)
  {
     ret++;
     ________;
  }
  return ret;
}
```

则空格内要填入的语句是（ ）。
A. x >>= 1
B. x &= x - 1
C. x |= x >> 1
D. x <<= 1

答案：B
解析：排除法+模拟

二 、 不定 项选择题（共 5 题，每题 2 分，共计 10 分 ；每题有一个或多个正确选
项，多选或少选均不得分 ）

1. NOIP 初赛中，选手可以带入考场的有（ ）。
A. 笔
B. 橡皮
C. 手机（关机）
D. 草稿纸

答案：AB

解析：草稿纸是不允许带的

2. 2-3 树是一种特殊的树，它满足两个条件：
（1）每个内部结点有两个或三个子结点；
（2）所有的叶结点到根的路径长度相同。
如果一棵 2-3 树有 10 个叶结点，那么它可能有（ ）个非叶结点。
A. 5
B. 6
C. 7
D. 8

答案：CD
解析：把最底层的节点都画出来向上连，先在符合条件的前提下把每 2 个点连到一起，这样发现有 8 个点，可以发现这是上界。同理把尽量多的 3 个点连在一起，答案是 7，可以发现这是下届，故选 CD

3. 下列关于最短路算法的说法正确的有（ ）。
A. 当图中不存在负权回路但是存在负权边时，Dijkstra 算法不一定能求出源点到所有点的最短路。
B. 当图中不存在负权边时，调用多次 Dijkstra 算法能求出每对顶点间最短路径。
C. 图中存在负权回路时，调用一次 Dijkstra 算法也一定能求出源点到所有点的最短路。
D. 当图中不存在负权边时，调用一次 Dijkstra 算法不能用于每对顶点间最短路计算。

答案：ABD

解析：dijstra 算法不适用于负权图，而且它用于求单点到其他点的最短路。熟悉图论的同学这是一个常识题。Dijkstra 是单源最短路算法，因此多次调用必然能够得到所有点对的最短路。如果图中出现负环，那么 dijkstra 算法就会在这个环里不断地转，因此无法求出最短路。同理负权。

4. 下列说法中，是树的性质的有（ ）。
A. 无环
B. 任意两个结点之间有且只有一条简单路径
C. 有且只有一个简单环
D. 边的数目恰是顶点数目减 1

答案：ABD
解析：树的基本知识

5. 下列关于图灵奖的说法中，正确的有（ ）。
A. 图灵奖是由电气和电子工程师协会（IEEE）设立的。
B. 目前获得该奖项的华人学者只有姚期智教授一人。
C. 其名称取自计算机科学的先驱、英国科学家艾伦·麦席森·图灵。
D. 它是计算机界最负盛名、最崇高的一个奖项，有“计算机界的诺贝尔奖”之称。

答案：BCD
解析：图灵奖是 ACM 设立的

三、问题求解（共 2 题，每题 5 分，共计 10 分）

1. 甲乙丙丁四人在考虑周末要不要外出郊游。
已知①如果周末下雨，并且乙不去，则甲一定不去；②如果乙去，则丁一定去；③如果丙去，则丁一定不去；④如果丁不去，而且甲不去，则丙一定不去。如果周末丙去了，则甲_____（去了/没去）(1 分)，乙_____（去了/没去）(1 分)，丁_____（去了/没去）(1 分)，周末_____（下雨/没下雨）(2 分)。

2. 方程 a*b = (a or b) * (a and b)，在 a,b 都取 [0, 31] 中的整数时，共有_____组解。（*表示乘法；or 表示按位或运算；and 表示按位与运算）

1.
由于点数很小，手动模拟下。从条件 3 开始找，即可找到答案。
2.
3.
首先如果 b 是 a 的子集，那么条件必然成立。然后手动简单玩一下，发现只有 1 位和 2 位情况存在特例。手动找到这些的答案即可。
科学的解释是：设 a and b=x, a xor x=y, b xor x=z，则 (x+y)(x+z)=x(x+y+z)，即 yz=0，即 a and b=a 或 a and b=b
4.

5.

6.

四、阅读程序写结果（共 4 题，每题 8 分，共计 32 分）

1.

```
#include<cstdio>
int main() {
int x;
scanf("%d", &x);
int res = 0;
for (int i = 0; i < x; ++i) {
  if (i * i % x == 1) {
    ++res;
  }
}
printf("%d", res);
return 0;
}
```

输入：15

输出：4

解析：简单模拟即可。这种题非常需要细心、耐心。

2.

```
#include<cstdio>
int n, d[100];
bool v[100];
int main() {
scanf("%d", &n);
for (int i = 0; i < n; ++i) {
  scanf("%d", d + i);
  v[i] = false;
}
int cnt = 0;
for (int i = 0; i < n; ++i) {
  if (!v[i]) {
    for (int j = i; !v[j]; j = d[j]) {
      v[j] = true;
    }
    ++cnt;
  }
}
printf("%d\n", cnt);
return 0;
```

```
}
```
输入：10 7 1 4 3 2 5 9 8 0 6
输出：6

解析：可以看到是读入一个序列，每个点都有一个出度。可以发现这是在找环的个数，手动模拟或者画个图数一下都可以。

3.
```
#include<iostream>
using namespace std;
string s;
long longmagic(int l, int r) {
long long ans = 0;
for (int i = l; i <= r; ++i) {
  ans = ans * 4 + s[i] - 'a' + 1;
}
return ans;
}
int main() {
cin >> s;
int len = s.length();
int ans = 0;
for (int l1 = 0; l1 < len; ++l1) {
  for (int r1 = l1; r1 < len; ++r1) {
    bool bo = true;
    for (int l2 = 0; l2 < len; ++l2) {
      for (int r2 = l2; r2 < len; ++r2) {
        if (magic(l1, r1) == magic(l2, r2)&& (l1 != l2 || r1 != r2)) {
          bo = false;
        }
      }
    }
    if (bo) {
      ans += 1;
    }
  }
}
cout << ans << endl;
return 0;
}
```
输入：abacaba
输出：16

解析：magic(l,r)是对于 s[l,r]的字符串哈希，底下枚举了两个子串，那么答案其实就是不重复出现的子串个数，手动数一下就好了。

4.

```
#include<cstdio>
using namespace std;
const int N =110;
bool isUse[N];
int n, t;
int a[N], b[N];
bool isSmall() {
for (int i = 1; i <= n; ++i)
  if (a[i] != b[i]) return a[i] < b[i];
return false;
}
boolgetPermutation(int pos) {
if (pos > n) {
  return isSmall();
}
for (int i = 1; i <= n; ++i) {
  if (!isUse[i]) {
    b[pos] = i; isUse[i] = true;
    if (getPermutation(pos + 1)) {
      return true;
    }
    isUse[i] = false;
  }
}
return false;
}
void getNext() {
for (int i = 1; i <= n; ++i) {
  isUse[i] = false;
}
getPermutation(1);
for (int i = 1; i <= n; ++i) {
  a[i] = b[i];
}
}
int main() {
scanf("%d%d", &n, &t);
for (int i = 1; i <= n; ++i) {
  scanf("%d", &a[i]);
}
```

```
for (int i = 1; i <= t; ++i) {
  getNext();
}
for (int i = 1; i <= n; ++i) {
  printf("%d", a[i]);
  if (i == n) putchar('\n'); else putchar(' ');
}
return 0;
}
```

输入 1：6 10 1 6 4 5 32
输出 1：2 1 3 5 6 4（3 分）
输入 2：6 200 1 5 3 4 26
输出 2：3 2 5 6 1 4（5 分）

解析：可以发现这一大堆函数唯一的用处就是找到字典序大于他的下一个排列（其实看到输入都是全排列的元素、以及一个 next 大概就可以猜到了。），对于第一个询问可以手动找到下 10 个排列。对于第二个询问，可以把后几位带在一起算，每次看把某一位更新成下一个值需要加上多少。当然也可以直接进行康托展开对全排列进行计数。

五、完善程序（共 2 题，每题 14 分，共计 28 分）

1. 对于一个 1 到 n 的排列 p(即 1 到 n 中每一个数在 p 中出现了恰好一次)，令 qi 为第 i 个位置之后第一个比 pi 值更大的位置，如果不存在这样的位置，则 qi =n+1。
举例来说，如果 n=5 且 p 为 1 5 4 2 3，则 q 为 2 6 6 5 6。
下列程序读入了排列 p，使用双向链表求解了答案。试补全程序。（第二空 2 分，其余 3 分）
数据范围 1 ≤ n ≤ 105。

```
#include<iostream>
using namespace std;
const int N =100010;
int n;
int L[N], R[N],a[N];
int main() {
cin >> n;
for (int i = 1; i <= n; ++i) {
  int x;
  cin >> x;
  a[x] = i ;
}
for (int i = 1; i <= n; ++i) {
  R[i]= i + 1;
  L[i] = i - 1;
```

```
}
for (int i = 1; i <= n; ++i) {
  L[R[a[i]]] = L[a[i]];
  R[L[a[i]]] = R[a[i]];
}
for (int i = 1; i <= n; ++i) {
  cout <<R[i]<< " ";
}
cout << endl;
return 0;
}
```

解析：

1. 一个有点鬼畜的双向链表。对于第一个空我们会发现如果 a[i]=x 的话直接 cin>>a[i]其实就好了，所以必然是 a[x]=i，意思是 x 是在第几个位置。

2,3,4 空 可以考虑仿写。完善程序出现“复读机”套路是很常见的。当然不能全部复读，要讲究对称性。

5 空 我们发现题目既然要求第 i 个数后面最近的一个比他大的,那么必然是 i 的后继，即 R[i]。

当然已经做出前几空了把题面的那组数据带进去就会发现的确是 R[i]。

如果原理不懂得可以画图进行模拟。不过反正写题本身很简单。

2. 一只小猪要买 N 件物品(N 不超过 1000)。

它要买的所有物品在两家商店里都有卖。第 i 件物品在第一家商店的价格是 a[i]，在第二家商店的价格是 b[i]，两个价格都不小于 0 且不超过 10000。如果在第一家商店买的物品的总额不少于 50000，那么在第一家店买的物品都可以打 95 折（价格变为原来的 0.95 倍）。

求小猪买齐所有物品所需最少的总额。

输入：第一行一个数 N。接下来 N 行，每行两个数。第 i 行的两个数分别代表 a[i]，b[i]。

输出：输出一行一个数，表示最少需要的总额，保留两位小数。

试补全程序。（第一空 2 分，其余 3 分）

```
#include<cstdio>
#include<algorithm>
using namespace std;
const int Inf =1000000000;
const int threshold = 50000;
const int maxn =1000;
int n, a[maxn],b[maxn];
bool put_a[maxn];
int total_a,total_b;
double ans;
int f[threshold];
int main() {
```

```
scanf("%d", &n);
total_a = total_b = 0;
for (int i = 0; i < n; ++i) {
  scanf("%d%d", a + i, b + i);
  if (a[i] <= b[i]) total_a += a[i];
  else total_b += b[i];
}
ans = total_a + total_b;
total_a = total_b = 0;
for (int i = 0; i < n; ++i) {
  if( a[i] * 0.95<= b[i] ){
    put_a[i] = true;
    total_a += a[i];
  } else {
    put_a[i] = false;
    total_b += b[i];
  }
}
if(total_a >= threshold){
  printf("%.2f", total_a * 0.95 +total_b);
  return 0;
}
f[0] = 0;
for (int i = 1; i < threshold; ++i)
  f[i] = Inf;
int total_b_prefix = 0;
for (int i = 0; i < n; ++i)
if (!put_a[i]) {
  total_b_prefix += b[i];
  for (int j = threshold - 1; j >= 0;--j) {
    if (total_a +j + a[i] >= threshold && f[j] != Inf)
        ans = min(ans, (total_a + j + a[i]) * 0.95+f[j] + total_b -
total_b_prefix);
    f[j] = min(f[j] + b[i], j >= a[i] ? f[j - a[i]] : Inf);
  }
}
printf("%.2f", ans);
return 0;
}
```

解析：

首先先考虑算法：如果第一家便宜肯定选第一家。

（以下内容感谢知乎@林泽辉 指正）

我们考虑程序后半段的思路，大致就是假设能够在第一家买够 50000，最少花多少钱。

那么如果打过九五折第一家更优，那么必然会选择第一家。反正剩下的物品都是中间物品。
那剩下的一些呢？我们称为“中间物品”。尽可能选择其中一些“中间物品”在 A 买，凑足 50000 元，使得比 B 便宜，但是尽可能的少。这是一个背包问题。
实现上，f[i,j]表示前 i 个物品，在额外在 A 店花了 j 元的情况下，购买 B 店“中间物品”的最小值。i 呢？滚动数组空间降维。
考虑为什么要先进行这个贪心，如果直接进行正常的背包的话，背包的大小将会是 物品个数 * 物品大小 ，如果先进行贪心的话，背包的大小会减小到 50000. 这样就可以接受了。

第一空：如果直接 a[i]<=b[i]的话上文就算过了，没必要单独循环一次。考虑贪心那么必然是看加了优惠之后 a[i]是否优于 b[i]；

第二空：考虑 printf 里面的部分，那么如果 a 的总和已经满足优惠，直接优惠掉即可；

第三空：考虑仿写 min 里面的部分，那么肯定是当前枚举的优惠幅度超过了 50000。这是考虑如果我们要买第 i 件，还额外花了 j 元在“中间物品”上的情况；

第四空：这是计算总价，第一店所有东西打折后，加上所有第二点需要购买的东西；

第五空：考虑它为啥要判 j>=a[i]，这个是做个背包问题都知道，这是背包问题的转移

第二十四届全国青少年信息学奥林匹克联赛初赛
提高组参考答案

一、单项选择题（共 10 题，每题 2 分，共计 20 分）

1	2	3	4	5	6	7	8	9	10
D	D	B	A	D	B	B	A	D	B

二、不定项选择题（共 5 题，每题 2 分，共计 10 分；每题有一个或多个正确选项，没有部分分）

1	2	3	4	5
AB	CD	ABD	ABD	BCD

三、问题求解（共 2 题，每题 5 分，共计 10 分）

1. 去了 没去 没去 没下雨 （第 4 空 2 分，其余 1 分）
2. 454

四、阅读程序写结果（共 4 题，每题 8 分，共计 32 分）

1. 4
2. 6
3. 16
4. 输出 1：2 1 3 5 6 4 （3 分） 输出 2：3 2 5 6 1 4 （5 分）

五、完善程序（共计 28 分，以下各程序填空可能还有一些等价的写法，由各省赛区组织本省专家审定及上机验证，可以不上报 CCF NOI 科学委员会复核）

		Pascal 语言	C++语言	C 语言	分值
1.	(1)	`a[x] := i`	`a[x] = i`		3
	(2)	`i + 1`			2
	(3)	`R[a[i]]`			3
	(4)	`a[i]`			3
	(5)	`R[i]`			3
2.	(1)	a[i] * 0.95 <= b[i] 或 b[i] >= a[i] * 0.95			2
	(2)	total_a >= threshold 或 threshold <= total_a 或 total_a >= 50000 或 50000 <= total_a			3
	(3)	total_a + j + a[i]			3
	(4)	f[j] + total_b - total_b_prefix			3
	(5)	f[j - a[i]]			3

北京高考在线是长期为中学老师、家长和考生提供新鲜的高考资讯、专业的高考政策解读、科学的升学规划以及实用的升学讲座活动等全方位服务的升学服务平台。自 2014 年成立以来一直致力于服务北京考生，助力千万学子，圆梦高考。

目前，北京高考在线拥有旗下拥有北京高考在线网站和北京高考资讯微信公众号两大媒体矩阵，关注用户超 20 万+。

北京高考在线_2020 年北京高考门户网站

http://www.gaokzx.com/

北京高考资讯微信：bj-gaokao