AtCoder Beginner Contest 020
解説

AtCoder株式会社 代表取締役
高橋 直大

競技プログラミングを始める前に

AtCoder

- 競技プログラミングをやったことがない人へ
  - まずはこっちのスライドを見よう!
  - http://www.slideshare.net/chokudai/abc004

A問題　クイズ

1.問題概要
2.アルゴリズム

A問題　問題概要

- 標準入力から整数 1 または 2 が与えられる
- 1 が与えられたら “ABC”
  2 が与えられたら “chokudai” と出力せよ
- （※ クイズの答えは問題ページの「出力例」の欄に書かれていました）

A問題　問題概要

AtCoder

- 文章での説明より、ソースコードを
お見せしてしまう方が簡単でしょう
- 3つの実装方針を C++ のコードで紹介します
- 他の言語でも、手続き型言語であれば
似たようなコードになるでしょう

AtCoder

- 方針 1: 条件分岐

```
int main(){
    int Q;
    cin >> Q;
    if (Q == 1){
        cout << "ABC" << endl;
    } else { // 1 でないなら 2
        cout << "chokudai" << endl;
    }
}
```

A問題　問題概要

- 方針 2: 三項演算子

```
int main(){
    int Q;
    cin >> Q;
    cout << (Q == 1 ? “ABC” : “chokudai”) << endl;
}
```

- (条件 ? A : B) は 条件 が真なら A, 偽なら B

A問題　問題概要

- 方針 3: 配列

```
int main(){
    string ans[2] = {“ABC”, “chokudai”};
    int Q;
    cin >> Q;
    cout << ans[Q - 1] << endl; // 配列の添字は 0 から
}
```

- 方針 1, 2, 3 のいずれかが他よりも好ましい、
ということは特にないでしょう (“レース”をしないなら)

B問題　足し算

1.問題概要
2.アルゴリズム

B問題　問題概要

AtCoder

- 整数 A, B が与えられる
- A と B の十進表記をこの順に連結して得られる整数を 2 倍したものを出力せよ
- 1 ≦ A, B ≦ 999

B問題　アルゴリズム

- まずは A と B の十進表記を連結する

- 整数が与えられているとはいえ、
  文字列として受け取ることが可能

- 多くの現代プログラミング言語では、
  文字列を '+' 演算子で連結できる

```
int main(){
    string A, B;
    cin >> A >> B;
    string AB = A + B;
    ...
```

B問題　アルゴリズム

AtCoder

- 次に、得られた文字列を整数に変換する

- 方法は言語によって異なるでしょう。
「<言語名> 文字列 整数 変換」などで検索

- 整数にしてしまえば 2 倍するのは簡単

```
    …
    cout << atoi(AB.c_str()) * 2 << endl;
}
```

- 文字列に変換せず、掛け算や割り算で解くことも可能です

C問題　壁抜け

1. 問題概要
2. アルゴリズム

C問題　問題概要

- 正方形のマスが縦H行、横W列に並んでおり、各マスは白か黒。スタート･ゴールが白マスにある
- スタートからゴールに T 秒以内に到着したい。上下左右に動けるが、白マスに移動するのに 1 秒、黒マスに移動するのに x 秒かかる
- 目標を達成できるような x の最大値は？

- $2 \leqq H, W \leqq 10, 2 \leqq T \leqq 10^9$
- 求める最大値が存在するような入力が与えられる（1度は黒マスを踏まないとゴールできない、かつ x = 1 なら間に合う）

C問題　アルゴリズム (部分点 1)

- T 秒以内にゴールできるような x の最大値を「直接」求めるのは難しそう
- x ≧ T のとき、T 秒以内のゴールは不可能（一度は黒マスを踏む必要があるような入力が来る）
- 「この x の値で T 秒以内にゴール可能か？」を x = T-1, T-2, … で調べ、最初に Yes となる x が答え

- このままでは満点 ($T \leqq 10^9$) は取れないが、とりあえず 70 点まではこの方針で

C問題　アルゴリズム (部分点 1)

- 部分点 1 (40 点): 2 ≦ H, W ≦ 3, 2 ≦ T ≦ 30

- ここまでマス目が狭いと、スタートからゴールまでの経路をすべて列挙したところで大して多くはない
- 「この x で T 秒以内にゴール可能か？」の判定のため DFS（深さ優先探索）などで経路をすべて列挙し、T 秒以内にゴールする経路が存在するかチェック

- 計算量は実装によるが、実行時間が問題になるような制約ではないはず

C問題　アルゴリズム (部分点 2)

- 部分点 2 (計 70 点): 2 ≦ H, W ≦ 10, 2 ≦ T ≦ 30

- もはや経路の全列挙は不可能。最短経路を効率的に求めるアルゴリズムが知られているので、それを使う
- ダイクストラ法、ベルマン-フォード法、ワーシャル-フロイド法、… (「最短路問題」などで検索)

- 今回はマス目の数が 100 以下と少ないため、T 回繰り返すことを考慮してもどれを使用しても可

(スクリプト言語でワーシャル-フロイド法は厳しいか)

C問題　アルゴリズム (満点)

- 満点 (100 点): $2 \leqq H, W \leqq 10$, $2 \leqq T \leqq 10^9$
- 「この x で T 秒以内にゴール可能か?」…☆ を x = T-1, T-2, … とすべて調べることはもはや不可能
- 本当にこの順にすべて調べる必要はあるだろうか?
- x を小さくすると、最短経路も短くなる。従って、
  x = i で ☆ の答えが Yes なら、x < i のときも Yes.
  x = i で ☆ の答えが No なら、x > i のときも No.

C問題　アルゴリズム (満点)

AtCoder

- lo = 1, hi = T として、hi − lo = 1 となるまで以下を繰り返すと、lo に求めたい最大値が入る
  mi = (lo + hi) / 2
  if(x = mi のときT秒以内にゴール可能) then lo = mi
  else hi = mi

- 1回のループで hi − lo がおよそ半分になるため、$T = 10^9$ でも $\log_2(10^9) = 30$ 周程度で終了する

- 「二分探索」と呼ばれる手法

- 別解もあります (x > H*W のときは黒を通る回数を最小化した上で白の回数を最小化するべきで、数式で最大値を出す。ダメなら x ≦ H*W のときを全部調べる)

D問題　LCM Rush

1. 問題概要
2. アルゴリズム

D問題　問題概要

- 整数 N, K が与えられる

- LCM(1, K) + LCM(2, K) + ⋯ + LCM(N, K) を 1,000,000,007 で割った余りを求めよ

  （LCM(a, b) は a と b の最小公倍数）

- $1 \leqq N, K \leqq 10^9$

D問題　アルゴリズム (部分点 1)

AtCoder

- 部分点 1 (5 点): 1 ≦ N, K ≦ 100

- i * K は i の倍数でも K の倍数でもあるから、LCM(i, K) は i * K 以下
- 1 から i * K までの各整数について、小さい順に i と K の両方で割っていき、最初に両方で割り切れた数が LCM(i, K)
- これを 1〜N のすべてについて行う

- 時間計算量 $O(N^2 K)$

D問題　アルゴリズム (部分点 2)

AtCoder

- 部分点 2 (計 15 点): $1 \leqq N \leqq 10^4$, $1 \leqq K \leqq 100$

- LCM(i, K) をもっと速く求めたい
- 「最小公倍数 プログラム」などで検索
- LCM(a, b) = a * b / GCD(a, b) という関係がある（GCD(a, b) は a と b の最大公約数））
- 「ユークリッドの互除法」というGCDを高速に計算する方法がある（計算量は O(log(2つのうち小さい方の数))）が、今回は 1～K を全部試しても可

- 時間計算量 O(N log(K)) または O(NK)

D問題　アルゴリズム (部分点 3)

- 部分点 3 (計 100 点): $1 \leqq N \leqq 10^9$, $1 \leqq K \leqq 100$

- もはや 1～N をすべて列挙することができない
- 本当に列挙する必要があるのだろうか？

- 前述の通り LCM と GCD には深い関係があるが、
その GCD には GCD(i, K) = GCD(i % K, K)
という性質がある
（i % K は i を K で割った余り。
「ユークリッドの互除法」はこれに基づく）

- K で割った余りで 1～N を分類したらどうか？

D問題　アルゴリズム (部分点 3)

- 分かりやすさのため具体例(N=53, K=10)で説明する

- 1～53 のうち 10 で割って 2 余る数は
  2, 12, 22, 32, 42, 52 の 6 個
- これらのどれを i としても GCD(i,10)=GCD(2,10)=2
- 従って、これらの各 i に対する LCM(i, 10) の和は
  2*10/2 + 12*10/2 + … + 52*10/2
  = $\underline{(2 + 12 + \cdots + 52)}$ * 10 / 2

- 下線部は等差数列の和で、O(1)で求まる
  （よく分からなければ検索。
  計算例: ((最初の項) + (最後の項)) * (項数) / ２ ）

D問題　アルゴリズム (部分点 3)

- 同様に、0〜K-1 のすべての整数 r について「1〜N のうち K で割った余りが r であるような整数 I すべてについての LCM(i, K) の和」を求めて合計する

- 時間計算量 O(K log(K))

D問題　アルゴリズム (満点)

- 満点 (101 点): $1 \leqq N, K \leqq 10^9$
- K も N と同程度に大きいと、1〜N を K で割った余りで 0〜K-1 に分類したところで計算量が減らない
- GCD(i, K) の値には相当数の重複が含まれないか？
- GCD(i, K) は K の約数の値しかとらず、
  $K \leqq 10^9$ のとき K の約数は高々 1344 個
  (K = 735134400 で最大)
- GCD(i, K) の値で 1〜N を分類したらどうか？

D問題　アルゴリズム (満点)

- 以下、N = 41, K = 12 の具体例で説明する

- まず、GCD(i, 12) = 1 なるすべての i について
  LCM(i, 12) を足しあわせたものを求めてみる

- LCM(1, 12) + LCM(5, 12) + LCM(7, 12) + …
  = 1 * 12 / 1 + 5 * 12 / 1 + 7 * 12 / 1 + …
  = $\underline{(1 + 5 + 7 + \cdots)}$ * 12 / 1

- 下線部の和を効率よく求めたい

D問題　アルゴリズム (満点)

- GCD(i, 12) = 1 なるすべての i の和を効率よく求めたい
- 1〜N のすべての整数の和から
  (1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 + 11 + 12 + 13 + …)

- 2 や 3 の倍数を引く（GCD(i, 12) = 1 を満たさない）
  - (2 + 4 + 6 + 8 + 10 + 12 + …)
  - (3 + 6 + 9 + 12 + …)

- すると 6 の倍数が二度引かれてしまうので足し直す
  +(+ 6 + 12 + …)

- ここまでの式はいずれも O(1) で計算可
  (1 + 2 + … + n = n(n+1)/2 を使うとよい)

D問題　アルゴリズム (満点)

- GCD(i, K) = 1 なる i すべてについての LCM(i, K) の和が求まったが、1 以外の K の約数についてはどうか

- 再び N = 41, K = 12 の具体例で考える。
GCD(i, 12) = 2 なる i (1 ≦ i ≦ 41) すべてについての LCM(i, 12) の和を求める

- GCD(i, 12) = 2 のとき、i = 2j (1 ≦ j ≦ 20) とおくと
GCD(i, 12) = GCD(2j, 12) = 2 * GCD(j, 6)
- 従って、「GCD(j, 6) = 1 なる j (1 ≦ j ≦ 20) すべてについての LCM(j, 6) の和」の 2 倍を求めればよく、GCD(i, K) = 1 のケースに帰着する

D問題　アルゴリズム (満点)

- 同様に、K の約数 d すべてについて「GCD(i, K) = d なる整数 i (1 ≦ i ≦ N) すべてについての LCM(i, K) の和」を求めて合計する

- 各 d について「K/d の素因数全体の集合の部分集合すべて」を列挙するので、時間計算量を大雑把に見積もると、f(n) を n の約数の個数、g(n) を素因数の個数として $O(f(K) * 2^{g(K)} * g(K))$

- $K \leqq 10^9$ のとき $f(K) \leqq 1344$, $g(K) \leqq 9$

- 実際にはより少ない計算量で求まる

D問題　おわりに

- お疲れ様でした。

- この問題は AtCoder Beginner Contest としては規格外の難易度で、AtCoder Regular Contest の問題 D の標準レベル程度でしょう。
（が、典型的ではあると思います）
満点がとれた人は Beginner とは呼びがたいです。