Editorial prova iniciante

Escrito por Gustavo Nunes e Yuri Cardoso

A - Divisão do terreno

Conhecimentos necessários:
- If

O problema basicamente dava um retangulo e iria fazer 2 cortes nele, um na horizontal e outro na vertical. O texto do problema assustou um pouco os competidores, pois era uma que utilizava apenas "if". A solução consiste em checar para cada ponto dado na entrada, qual era o quadrante em que ele se encontrava. Dadas as posições A e B das cercas, um ponto com coordenadas (x, y) está na parte superior direita se (x > A && y >= B), está na parte superior esquerda se (x <= A && y > B), está na parte inferior esquerda se (x < A && y <= B) e está na parte inferior direita se (x >= A && y < B).

B - Teclado

Conhecimentos necessários:
- Laços de repetição
- Logartimos e bases numéricas (opcional)

Solução 1:

Comece com duas variávis x = 1, e c = 0, e faça x*=2 enquanto x < n, somando 1 em c a cada iteração.

Se após as multiplicações x for igual a n, printe c, caso contrário printe -1.

Solução 2:

Se n for uma potencia de 2, printe o logaritmo de n na base 2, caso contrário printe -1.

C - Peça desconhecida

Conhecimentos necessários:
- Nenhum

Sabendo que o volume de um cilindro de raio R e altura H é 3.141 * R * R * H, a questão queria saber, em outras palavras, qual foi o aumento do volume ao adicionar a peça no cilindro. Para implementar isso é preciso saber qual o volume inicial V1 (V1 = 3.141 * R * R * H1), e o volume final V2 (V2 = 3.141 * R * R * H2), com isso, a resposta é a diferença entre o volume final (V2) e o volume inicial (V1).

D – Binário

Conhecimentos necessários:
- Bases numéricas

O problema se resumia em: “printe todos os numeros entre 0 e (2^n) – 1, na forma binária”.

E – Chuva

Conhecimentos necessários:
- If

Tal problema queria saber qual a posição do maior elemento lido (se era o primeiro, segundo ou terceiro valor lido). Como os 3 valores são distintos, é suficiente checar para cada um, se ele é maior do que os outros 2, pois sempre haverá uma resposta única.

F – Conta

Conhecimentos necessários:
- Nenhum

O problema F era o mais fácil da prova e era apenas calcular o resultado da conta "(A+B)*(B-A)*(C+D)*(C-D) + A + B + C + D" em alguma das linguagens permitidas na competição.

G – Senha

Conhecimentos necessários:
- Manipulação de strings
- Módulo (restos)

O problema se resume em basicamente girar a string k vezes para a direita. Porém, a cada n (n é o tamanho da string) iterações, a string resultante volta a ser a string original, portanto, era necessário girar a string apenas k%n (onde k%n indica o resto da divisão de k por n) vezes.
Seja x = k%n.
Como o problema só pedia para printar a string, bastava printar os ultimos x caracteres da string, e depois printar os n-x primeiros caracteres da string.

H – Fila

Conhecimentos necessários:
- Pilha

A estrutura descrita por Ariel era a Pilha, e o nome do problema (Fila) não representava nada. A forma mais simples de se resolver essa questão é ter um vetor, que irá representar sua pilha, e uma variável (indice) para indicar qual a posição do vetor que se deve inserir o próximo elemento. Inicialmente a variável indice é 0, pois não tem nenhum elemento ainda no seu conjunto de valores.

Para inserir um elemento, ele deve ser colocado na posição vetor[indice], e após isso deve-se fazer indice++, indo para a próxima posição livre.

Para remover um elemento, a única coisa que se deve fazer é diminuir o indice em 1 (indice--), se a variável for maior do que 0, caso contrário não faça nada pois não tem elemento a ser removido.

Para consultar o último valor inserido, deve-se fazer printf("%d\n", vetor[indice-1]), se indice > 0, caso contrário deve imprimir na tela o -1, pois não existe elemento no conjunto atualmente.

I - Prédios

Conhecimentos necessários:
- Estruturas (structs)
- Ordenação

Vamos separar a resolução em duas partes:

Parte 1: assuma que as coomeias estão ordenadas na ordem que batmel irá recolhê-las. Se isso acontecer, então basta percorrer todas as colmeias, e a cada vez que você for para uma colmeia, some abs(y – y_anterior) à resposta, onde y é a coordenada y da colmeia atual, e y_anterior é a coordenada y da colmeia anterior, e abs(x) é o valor absoluto de x.

Porém, o problema não garante que as colmeias serão dadas na ordem certa, portanto o problema se resume em ordenar as colmeias na ordem certa, e depois resolver da maneira indicada na Parte 1.

Parte 2: ordenar as colmeias.

Para ordenar as colmeias, basta criar uma função de comparação que compara duas structs colmeia da forma:

```
struct colmeia{
    int x, y, população;
}
```

Sejam duas struct A e B. A vem antes de B se:
A.x < B.x(B vem antes de A se B.x < A.x)
Se A.x == B.x, então deve-se comparar as populações:
Se A.população > B.população A vem antes de B(da mesma forma, B vem antes de A do contrario).
Se A.x == B.x e A.popuação == B.população, quem tiver o menor y vem primeiro.

Feita a ordenação, basta voltar à parte 1. Lembre-se de usar long long int na resposta, pois o valor final pode resultar em um número muito grande.

J - Discos voadores

Conhecimentos necessários:
- Fórmula para distância entre pontos

Dois discos não se tocam, nem se sobrepoem se a distância entre seus centros for maior do que a soma dos seus raios. Sabendo disso é possível resolver o problema checando para cada disco voador se ele está isolado. Um disco voador está isolado se

ele não toca, nem se sobrepoe a nenhum outro disco. A soluçao consiste de 2 laços de repetição 'for' aninhados.

K - Truco 2.0

Conhecimentos necessários:
- If

Tal problema era mais um de "if", mas que exigia uma leitura mais atenta do enunciado. O jogador 1 é considerado vencedor se TODAS as suas cartas forem maiores do que TODAS as cartas do jogador 2. O jogador 2 é considerado vencedor se TODAS as suas cartas forem maiores do que TODAS as cartas do jogador 1. Caso não aconteca nenhuma das situações acima, o jogo termina empatado.

L - Curso de digitação

Conhecimentos necessários:
- If

Muitos competidores tiveram o raciocínio certo, mas estavam escrevendo as palavras da saída errada, como "mediana" no lugar de "mediano", mas o sistema considera diferente as palavras "mediana" e "mediano" e retornava NO - Wrong Answer, mesmo a lógica estando certa. A solução do problema consiste em uma sequência de if's, verificando em qual intervalo o X da entrada estava. Segue abaixo a classificação de cada intervalo:
[1, 39] = devagar
[40, 80] = mediano
[81, 160] = veloz
[161, 1000] = inexistente