



Piscina C

Rush 00

Sumário: ESTE documento é o enunciado do módulo Rush 00 da Piscina C da 42.

Versão: 5.2

Conteúdo

I	Instruções	2
II	Preâmbulo	4
III	Enunciado Principal	6
IV	Rush 00	8
V	Rush 01	10
VI	Rush 02	11
VII	Rush 03	12
VIII	Rush 04	13
IX	Submissão e avaliação	14

Capítulo I

Instruções

- O grupo será automaticamente inscrito para defesa.
- No caso de ser cancelada, não terão direito a outra.
- Qualquer questão acerca do enunciado tem a consequência de o complicar.
- Deverá seguir o procedimento de entrega para o enunciado.
- O enunciado pode mudar até uma hora antes da entrega.
- O programa deverá compilar com as seguintes *flags*: -Wall -Wextra -Werror; e utiliza `cc`.
- Se o programa não compila, o grupo terá 0.
- O número de rush obrigatório do seu grupo seguirá esta regra:
O índice alfabético da primeira letra do login do líder da equipa (de 1 a 26) módulo 5.
- Deverá, portanto, executar o enunciado indicado com os seus parceiros impostos e apresentar-se para a defesa do projeto na hora marcada com todos os seus parceiros.
- O projeto deverá estar finalizado pela altura da defesa. O propósito da defesa é para que todos possam apresentar e explicar o projeto em todo o seu detalhe.
- Cada membro do grupo deve estar perfeitamente consciente do trabalho realizado. No caso de terem dividido o trabalho, certifiquem-se que todos perceberam o trabalho de cada um. Durante a defesa, serão feitas várias questões; a nota do grupo será baseada nas piores explicações.
- O contacto entre o grupo é da vossa responsabilidade. Têm todos os meios para entrar em contacto com os parceiros: telefone, carta, pombo correio, bola de cristal, etc. Não há necessidade de desculpas quanto ao funcionamento do grupo. A vida nem sempre é justa; é o que é.

- No entanto, se depois de tentar realmente tudo, um dos seus parceiros ainda estiver incomunicável: faça o rush de qualquer forma, e arranjaremos uma solução na defesa. Incluindo o caso da ausência do líder do grupo: todos possuem acesso ao repositório.
- Bom trabalho a todos.
- Se quiserem pontos bónus, podem submeter outros enunciados ou serem capazes de usar argumentos no programa para testar a vossa função.



Deve fazer o enunciado obrigatório de forma perfeita para apresentar os enunciados bónus: Se um enunciado bónus funcionar mas o original falhar os testes, o grupo terá 0.

Capítulo II

Preâmbulo

Here are the lyrics of a famous TV show for everyone:

[Verse 1]

I wanna be the very best
Like no one ever was
To catch them is my real test
To train them is my cause

I will travel across the land
Searching far and wide
Each pokemon to understand
The power that's inside

[Chorus]

Pokemon! Gotta catch 'em all! It's you and me
I know it's my destiny,
Pokemon! Oh you're my best friend
In a world, we must defend
Pokemon! A heart so true
Our courage will pull us through,

You teach me and I'll teach you,
Pokemon! Gotta catch'em all

[Chorus]


Every challenge along the way
With courage I will face.
I will battle every day
To claim my rightful place.
Come with me,
The time is right,
There's no better team.
Arm in arm we'll win the fight!
It's always been our dream!

[Chorus]

I could bet you were singing right now, but it doesn't matter for the moment. And this subject is not related to Pocket Monster by the way...

Capítulo III

Enunciado Principal

	Exercício : 00
	rush0X
	Pasta de entrega : <i>ex00/</i>
	Ficheiros para entregar : <code>main.c</code> , <code>ft_putchar.c</code> , <code>rush0X.c</code>
	Funções autorizadas : <code>write</code>

- Os arquivos que devem ser entregues são: `main.c`, `ft_putchar.c` e o seu `rush0X.c`, no qual 0X corresponde ao número do rush. Por exemplo, `rush00.c`.
- Estes três ficheiros serão compilados juntos.
- O ficheiro `ft_putchar.c` deve incluir a função `ft_putchar`.
- Exemplo do `main.c`:

```
int      main()
{
    rush(5, 5);
    return (0);
}
```

- Deverá, portanto, escrever a função `rush` tendo como parâmetro duas variáveis do tipo inteiro nomeadas respectivamente `x` e `y`.
- A função `rush` deverá exibir na tela um retângulo de `x` caracteres de largura, e `y` caracteres de altura.
- A sua função nunca deverá quebrar ou ficar em um ciclo infinito.
- O seu `main` será modificado na defesa para poder mudar os parâmetros da chamada à função `rush`. Aqui está um exemplo do que será testado:

```
int    main()
{
    rush(123, 42);
    return (0);
}
```


Capítulo IV

Rush 00

- `rush(5,3)` deverá mostrar:

```
$> ./a.out
o---o
|   |
o---o
$>
```

- `rush(5, 1)` deverá mostrar:

```
$> ./a.out
o---o
$>
```

- `rush(1, 1)` deverá mostrar:

```
$> ./a.out
o
$>
```

- `rush(1, 5)` deverá mostrar:

```
$> ./a.out
o
|
|
|
o
$>
```

- `rush(4, 4)` deverá mostrar:

```
$> ./a.out  
o--o  
|  |  
|  |  
o--o  
$>
```

Capítulo V

Rush 01

- `rush(5,3)` exhibirá:

```
$> ./a.out
/***\
*   *
\***/
$>
```

- `rush(5, 1)` deverá mostrar:

```
$> ./a.out
/***\
$>
```

- `rush(1, 1)` deverá mostrar:

```
$> ./a.out
/
$>
```

- `rush(1, 5)` deverá mostrar:

```
$> ./a.out
/
*
*
*
\
$>
```

- `rush(4, 4)` deverá mostrar:

```
$> ./a.out
/**\
*  *
*  *
\**/
$>
```

Capítulo VI

Rush 02

- `rush(5,3)` deberá mostrar:

```
$> ./a.out
ABBBA
B  B
CBBBC
$>
```

- `rush(5, 1)` deberá mostrar:

```
$> ./a.out
ABBBA
$>
```

- `rush(1, 1)` deberá mostrar:

```
$> ./a.out
A
$>
```

- `rush(1, 5)` deberá mostrar:

```
$> ./a.out
A
B
B
B
C
$>
```

- `rush(4, 4)` deberá mostrar:

```
$> ./a.out
ABBA
B  B
B  B
CBBC
$>
```

Capítulo VII

Rush 03

- `rush(5,3)` deberá mostrar:

```
$> ./a.out
ABBBC
B  B
ABBBC
$>
```

- `rush(5, 1)` deberá mostrar:

```
$> ./a.out
ABBBC
$>
```

- `rush(1, 1)` deberá mostrar:

```
$> ./a.out
A
$>
```

- `rush(1, 5)` deberá mostrar:

```
$> ./a.out
A
B
B
B
A
$>
```

- `rush(4, 4)` deberá mostrar:

```
$> ./a.out
ABBBC
B  B
B  B
ABBBC
$>
```

Capítulo VIII

Rush 04

- `rush(5,3)` deberá mostrar:

```
$> ./a.out
ABBBC
B  B
CBBBA
$>
```

- `rush(5, 1)` deberá mostrar:

```
$> ./a.out
ABBBC
$>
```

- `rush(1, 1)` deberá mostrar:

```
$> ./a.out
A
$>
```

- `rush(1, 5)` deberá mostrar:

```
$> ./a.out
A
B
B
B
C
$>
```

- `rush(4, 4)` deberá mostrar:

```
$> ./a.out
ABBC
B  B
B  B
CBBA
$>
```

Capítulo IX

Submissão e avaliação

Entrega o teu trabalho no teu repositório `Git`, como habitual. Apenas o trabalho dentro do teu repositório será avaliado durante a defesa. Não hesites em confirmar os nomes dos teus ficheiros para ter a certeza que estão corretos.

Como estes trabalhos não são avaliados por um programa, sinta-se livre de organizar os ficheiros como preferir, desde que entregue os ficheiros obrigatórios e que cumpra com os requisitos.



Apenas precisas de entregar os ficheiros pedidos no enunciado deste projeto.