
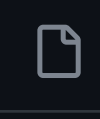
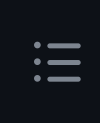


main1 branch0 tags

Go to fileAdd fileCode

 zuilpirola	Create resolucao.c	fcee9e7 · 2 days ago · 14 commits
 README.md	Update README.md	last month
 resolucao.c	Create resolucao.c	2 days ago

README.md

UNIVERSIDADE LUSÓFONA DE HUMANIDADES E TECNOLOGIAS

Linguagens de Programação I

Exercício Individual 3 - Fila do Restaurante

Na resolução deste exercício deve ser utilizada a Linguagem de Programação C. Para além da correta implementação dos requisitos, tenha em conta os seguintes aspetos:

- O código apresentado deve ser bem indentado.
- O código deve compilar sem erros ou *warnings* utilizando o *gcc* com as seguintes flags:
`g -Wla -Wall -Wpedantic -Wextra -Wdeclaration-after-statement`
- Tenha em atenção os nomes dados das variáveis, para que sejam indicadores daquilo que as mesmas vão conter.
- O trabalho deve ser desenvolvido e submetido de forma individual.

Este exercicio deverá ser submetido na plataforma Pandora até às 23h59 de dia XX de Abril e será contabilizado para a nota final da unidade curricular de acordo com os critérios disponibilizados na página da disciplina, concretamente nos slides da primeira aula.

Todos os trabalhos serão comparados utilizando um sistema de deteção de plágio.

Contexto

Um novo restaurante vegano estar a fazer sucesso em Lisboa. O problema é que não há a possibilidade de fazer reservas e a fila para conseguir uma mesa pode ser um desafio. Pensando nisso o Sr. Martins, dono do restaurante, pediu para que os alunos da disciplina elaborassem um programa que fizesse o controlo da fila de pessoas num dia de funcionamento normal do restaurante (14h-23h). O restaurante possui apenas 4 mesas (as mesas possuem numero ilimitado de pessoas) e estas nunca são partilhadas entre grupos distintos. Assim, quando as mesas do restaurante estão todas ocupadas, um grupo que acabe de chegar deverá aguardar na fila até que chegue a sua vez. Apesar de ainda não existir um programa para gerir a fila, o Sr. Martins tem um controlo significativo do restaurante. Ele informou-nos que nenhuma mesa fica ocupada mais do que 2h, pelo mesmo grupo. Após duas horas uma equipa vai até a mesa e limpa-a para receber um novo grupo. Há um funcionário que irá operar o programa e que terá informações sobre quantas pessoas chegaram e que horas chegaram. Com essas duas informações o programa deverá funcionar e atribuir automaticamente as 4 mesas às pessoas que chegam. O programa termina quando a hora de funcionamento ultrapassar a hora de fecho do restaurante >23.

Descrição

O ecrã de controlo do programa deve ter os seguintes campos:

```
-----
Fila:3-7-
Hora Atual: 15
-----

Mesa 1: 1
Mesa 2: 2
Mesa 3: 3
Mesa 4: 2

Quantas pessoas chegaram?
1
Que horas chegaram?
15
```

A tela de controlo acima sempre vai demonstrar quantas pessoas estão alocadas em uma mesa, sendo `@` o status que a mesa está disponível. É importante também ter o controlo de que horas a mesa iniciou para que ela fique disponível após 2h.

A variável `Hora Atual` é uma variável de controlo de tempo do restaurante que sempre é atualizada a partir do tempo em que novos grupos chegam.

O operador pode por exemplo dizer que 4 pessoas chegaram as 14h:

```
Quantas pessoas chegaram?
4
Que horas chegaram?
14
```

E em seguida pode dizer que 1 pessoa chegou as 15h:

```
Quantas pessoas chegaram?
1
Que horas chegaram?
15
```

Neste momento o horário do restaurante deve mudar de 14h para 15h. O operador do programa também pode atualizar o horário sem atribuição de grupo:

```
Quantas pessoas chegaram?
0
Que horas chegaram?
16
```

Para este caso ninguém será atribuído a nenhuma mesa ou fila e a hora do restaurante muda para 16h.

O operador digita apenas horas cheias.

A gestão da fila é feita da seguinte forma `Fila:3-7-` onde `-` é a separador entre grupos.

A hora lida sempre tem que ser maior ou igual a hora atual. Se isso não ocorrer basta apenas perguntar novamente `Que horas chegaram?`

Toda mesa tem em sua estrutura uma `ocupação` e uma `hora de início`. A hora de início é sempre a hora em que a mesa é iniciada com o grupo, independente da hora que ele entrou a fila.

Utilizem estrutura de dados tipo fila para controlar a fila do restaurante. Trabalhem com funções em separado para organizar melhor o código(Ex. Função que imprime, Função que faz a manutenção...)

Exemplo de execução:

```
-----
Fila:
Hora Atual: 14
-----

Mesa 1: 0
Mesa 2: 0
Mesa 3: 0
Mesa 4: 0

Quantas pessoas chegaram?
3
Que horas chegaram?
14

-----
Fila:
Hora Atual: 14
-----

Mesa 1: 3
Mesa 2: 0
Mesa 3: 0
Mesa 4: 0

Quantas pessoas chegaram?
99
Que horas chegaram?
14

-----
Fila:
Hora Atual: 14
-----

Mesa 1: 3
Mesa 2: 99
Mesa 3: 0
Mesa 4: 0

Quantas pessoas chegaram?
13
Que horas chegaram?
15

-----
Fila:
Hora Atual: 15
-----

Mesa 1: 3
Mesa 2: 99
Mesa 3: 13
Mesa 4: 0

Quantas pessoas chegaram?
5
Que horas chegaram?
16

-----
Fila:
Hora Atual: 16
-----

Mesa 1: 5
Mesa 2: 0
Mesa 3: 13
Mesa 4: 0

Quantas pessoas chegaram?
10
Que horas chegaram?
16

-----
Fila:
Hora Atual: 16
-----

Mesa 1: 5
Mesa 2: 10
Mesa 3: 13
Mesa 4: 0

Quantas pessoas chegaram?
10
Que horas chegaram?
16

-----
Fila:
Hora Atual: 16
-----

Mesa 1: 5
Mesa 2: 10
Mesa 3: 13
Mesa 4: 10

Quantas pessoas chegaram?
4
Que horas chegaram?
16

-----
Fila: 4-
Hora Atual: 16
-----

Mesa 1: 5
Mesa 2: 10
Mesa 3: 13
Mesa 4: 10

Quantas pessoas chegaram?
10
Que horas chegaram?
16

-----
Fila: 4-10-
Hora Atual: 16
-----

Mesa 1: 5
Mesa 2: 10
Mesa 3: 13
Mesa 4: 10

Quantas pessoas chegaram?
0
Que horas chegaram?
17

-----
Fila: 10-
Hora Atual: 17
-----

Mesa 1: 5
Mesa 2: 10
Mesa 3: 4
Mesa 4: 10

Quantas pessoas chegaram?
0
Que horas chegaram?
23
```

O programa termina.

Sugestão de funções para gerir a fila

```
//Variaveis da fila
int fila[MAX];
//endereco de final
int fim = 0;
//endereco de inicio
int ini = 0;

//Funcao de adicao fila
void qstore(int quant){
    //condicao para fila cheia
    if (fim == MAX){
        // printf("Fila cheia.\n");
        return;
    }
    //salva o valor no vetor
    fila[fim] = quant;
    //ajusta o final da fila
    fim++;
}


//funcao de extracao da fila
int qretrieve(){
    //condicao para lista vazia
    if (ini==fim){
        // printf("Fila vazia.\n");
        return NULL;
    }
    //mudar o inicio da fila
    ini++;
    //retornar o valor
    return fila[ini-1];
}
```


Honestidade Académica


Nesta disciplina, espera-se que cada aluno siga os mais altos padrões de honestidade académica. Trabalhos que sejam identificados como cópias serão anulados e os alunos envolvidos terão nota zero - quer tenham copiado, quer tenham deixado copiar. Para evitar situações deste género, recomendamos aos alunos que nunca partilhem ou mostrem o seu código. A decisão sobre se um trabalho é uma cópia cabe exclusivamente aos docentes da unidade curricular. Os alunos são encorajados a discutir os problemas com outros alunos mas não deverão, no entanto, copiar códigos, documentação e relatórios de outros alunos. Em nenhuma circunstância deverão partilhar os seus próprios códigos, documentação e relatórios. De facto, não devem sequer deixar códigos, documentação e relatórios em computadores de uso partilhado.


About

No description, website, or topics provided.

 Readme

 0 stars

 2 watching

 0 forks

Report repository

Releases

No releases published

Packages

No packages published

Contributors 2

 zuilpirola zuilpirola

 dsilveira32

Languages

