Instituto Superior de Engenharia de Coimbra

Licenciatura em Engenharia Informática – 19/20

Programação

4B: Outras Operações com Ficheiros

Francisco Pereira (xico@isec.pt)

Operações diretas em ficheiros

Instituto Superior de Engenharia de Coimbra

- Algumas operações que podem ser efetuadas

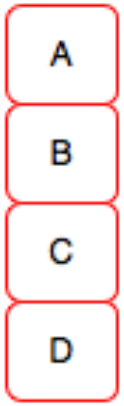
1. Consultar a informação armazenada
   - Listar / Pesquisar
2. Alterar a informação armazenada
   - Atualizar/Adicionar/Eliminar

Operações diretas em ficheiros

- Utilizar ficheiro binário de clientes do banco como exemplo
  - Versão sem valor inteiro no início

1. Listar todos os clientes
2. Atualizar montante de um cliente
3. Adicionar um novo cliente

banco.dat




```
typedef struct dados cliente;
struct dados {
   char nome[100];
   char nconta[15];
   int montante;
};
```

Operação 1: Listar todos os clientes

Instituto Superior de Engenharia de Coimbra

```
void listarFicheiro(char *nome){
    FILE *f;
    cliente c;

    f = fopen(nome, "rb");
    if(f==NULL)
        return;

    while(fread(&c, sizeof(cliente), 1, f) == 1)
        escreve_info(c);

    fclose(f);
}
```

Operação 2: Atualizar o saldo de um cliente

```
int atualizaSaldoV1(char *nome, char *conta, int valor);
```

- Atualiza o saldo da conta *conta* armazenada no ficheiro binário *nome*
- Devolve sucesso (1) ou falhanço (0)

- Situação a considerar:
  - É uma operação que necessita que o ficheiro seja aberto para leitura e escrita

Operação 2: Atualizar o saldo de um cliente

- Estratégia A
  - Utilizar um ficheiro temporário para onde é copiada a informação atualizada
  - No final
    - Apagar o ficheiro original.
    - Mudar o nome do ficheiro temporário.

```
int remove(char* nome);

int rename(char* antigo, char* novo);
```

Devolvem 0 se tudo correr bem

Operação 2A: Atualizar o saldo de um cliente

```
int atualizaSaldoV1(char *nome, char *conta, int valor){
    FILE *f, *g;
    cliente c;
    int flag = 0;

    f = fopen(nome, "rb");
    if(f == NULL) return 0;
    g = fopen("temp.dat", "wb");
    if(g == NULL){ fclose(f); return 0;
    }
    while(fread(&c, sizeof(cliente), 1, f) == 1){
        if(strcmp(c.nconta, conta) == 0){
            c.montante += valor;
            flag = 1;
        }
        fwrite(&c, sizeof(cliente), 1, g);
    }
    fclose(f); fclose(g);
    remove(nome); rename("temp.dat", nome);
    return flag;
}
```

Operação 2: Atualizar o saldo de um cliente

Instituto Superior de Engenharia de Coimbra

- Estratégia B
  - Abrir um ficheiro para leitura e escrita
    - Modos: r+, w+ ou a+
  - Trocar de leitura para escrita
    - Reposicionar o acesso
  - Trocar de escrita para leitura
    - Reposicionar o acesso
    - Esvaziar o buffer de output

Funções de Reposicionamento

Devolve a posição actual (em bytes) no ficheiro

```
long ftell(FILE *f);


int fseek(FILE *f, long offset, int origin);
```

Altera o ponto de acesso ao ficheiro (devolve 0 se correr bem)

SEEK_CUR: posição actual
SEEK_END: fim do ficheiro
SEEK_SET: início do ficheiro

Operação 2B: Atualizar o saldo de um cliente

```
int atualizaSaldoV2(char *nome, char *conta, int valor){
    FILE *f;
    cliente c;

    f = fopen(nome, "rb+");
    if(f == NULL)
        return 0;

    while(fread(&c, sizeof(cliente), 1, f) == 1){
        if(strcmp(c.nconta, conta) == 0){
            c.montante += valor;
            fseek(f, -sizeof(cliente), SEEK_CUR);
            fwrite(&c, sizeof(cliente), 1, f);
            fclose(f);
            return 1;
        }
    }
    fclose(f);
    return 0;
}
```

Operação 3: Adicionar um novo cliente

```
int adicionaCliente(char *nomeFich, cliente novo);
```

- Qual a melhor estratégia?
  1. Usar um ficheiro temporário
  2. Abrir o ficheiro para leitura/escrita

- Restrição importante
  - O ficheiro está ordenada por número de conta e deve permanecer ordenado depois da adição do novo cliente