ESTGOH
Escola Superior de Tecnologia e Gestão de Oliveira do Hospital
Instituto Politécnico de Coimbra

Engenharia Informática

Estruturas de Dados

2019/2020

Relatório

Tabelas, Ficheiros, Estruturas e Listas ligadas

António Pereira Nº 2019141051

David Ferreira Nº 2019141281

27/06/2020

Índice

1 Introdução

Nesta última ficha, pretende-se criar um programa em que o utilizador consiga inserir as máquinas de uma empresa e os dados de cada máquina, como por exemplo, a temperatura, a humidade, a energia consumida, etc. Estes dados tem que ser guardados numa lista ligada para cada máquina e todas estas listas tem que ficar guardadas numa estrutura. Ao longo deste relatório também iremos explicar os exercícios que nos foram propostos pela professora com prints dos códigos e outputs dos mesmos para se tornar mais simples de se perceber.

2 Trabalho realizado

2.1 Observando o esquema, crie as estruturas que considerar necessárias para guardar todos os dados indicados numa base de dados

Neste primeiro exercício, foi-nos pedido para criarmos estruturas que guardassem todos os dados indicados numa base de dados de uma empresa. Para isso, criamos uma lista ligada e uma estrutura. A lista ligada, vai guardar todos os dados de cada máquina que são: temperatura, data, Co2, humidade, produção e energia. Já a estrutura vai guardar os nomes de cada máquina e esta estrutura vai apontar para a lista dos dados, para deste modo saber os dados que cada máquina tem. A estrutura e a lista ligada são as seguintes:

```
typedef struct dataa *plista;
typedef struct dataa{
    char data[200];
    float Co2;
    float humidade;
    float producao;
    float temperatura;
    float energia;
    plista prox;
}dados;
```

Figura 1- Lista dos dados das máquinas

```
typedef struct maquina{
    char nome[200];
    plista pprox;
}nomesmaquinas;
```

Figura 2- Estrutura onde vai ser guardado o nome de cada máquina

2.2 Desenvolva os subprogramas que considerar adequados para realizar as seguintes operações

a) Na impossibilidade de comunicar com cada uma das máquinas, deve ser possível inserir manualmente os dados indicados, para cada máquina. No momento de uma falha, os valores registados na lista são 0's, incluindo a data.

Na alínea a criamos duas funções, uma para inserir os nomes das máquinas e outra para inserir os dados da máquina. Na função de inserir os nomes das máquinas, primeiro perguntamos ao utilizador para inserir quantas máquinas é que pretende e depois fazemos um for e dentro deste pedimos ao utilizado para inserir o nome de cada máquina que ele inseriu inicialmente. Ainda dentro deste for, o ponteiro pprox vai ser inicializado a null a cada nome inserido, pois, este ponteiro vai apontar para a lista dos dados, ou seja, cada máquina vai apontar para os seus dados. Já para inserir os dados da máquina, primeiro pedimos ao utilizador para inserir o nome da máquina onde quer inserir os dados e para ver se existe essa máquina fazemos um strcmp entre a string onde estão guardados os nomes e nome que o utilizador inseriu. Se for igual (0) então vão ser inseridos os dados dessa máquina. Ainda dentro deste if a informação da data vai ser copiada para uma variável auxiliar, depois esta variável secundaria vai ser igualda a /0, pois é uma string, e por último o conteúdo desta variável vai ser copiada para a variável das datas. Por fim, o no->prox, ou seja, o próximo elemento, vai apontar para o pprox que é para o no->prox ficar com os dados do ponteiro pprox e depois este vai ficar com os dados do no.
As duas funções são as seguintes:

```
int InsereMaquinas(nomesmaquinas ArrayNomes[200], int count)
{
    int qnt;
    printf("\nInsira o numero de maquinas que pretende:");
    scanf("%d", &qnt);
    for(int i=0;i<qnt;i++)
    {
        printf("\nInsira o nome da %d maquina:\n", i+1);
        fflush(stdin);
        gets(ArrayNomes[count].nome);
        ArrayNomes[count].pprox = NULL;
        count++;
    }
    return count;
}
```

Figura 3- Função para inserir os nomes das máquinas

```c
void InsereDados(int count, nomesmaquinas ArrayNomes[200])
{
    char maquina[100];
    time_t data;
    char DataAux[200];
    plista no = (plista)malloc(sizeof(dados));
    time(&data);
    printf("\nInsira o nome da maquina onde quer colocar os dados: ");
    fflush(stdin);
    gets(maquina);
    if(no==NULL)
        printf("\nNao ha memoria disponivel!");
    else {
        for (int i = 0; i < count; i++)
        {
            if(strcmp(ArrayNomes[i].nome, maquina)==0)
            {
                printf("\nInsira a quantidade de Co2 da maquina: ");
                scanf("%f", &no->Co2);
                printf("\nInsira a quantidade de energia produzida pela maquina: ");
                scanf("%f", &no->energia);
                printf("\nInsira a temperatura da maquina: ");
                scanf("%f", &no->temperatura);
                printf("\nInsira a quantidade produzida pela maquina: ");
                scanf("%f", &no->producao);
                printf("\nInsira a humidade da maquina atualmente: ");
                scanf("%f", &no->humidade);
```

Figura 4- Função para inserir os dados das máquinas

```c
                strcpy(DataAux, ctime(&data));
                DataAux[24] = '\0';
                strcpy(no->data, DataAux);
                no->prox = ArrayNomes[i].pprox;
                ArrayNomes[i].pprox = no;
            }
        }
    }
}
```

Figura 5- Continuação da função

Exemplo de output da função de inserir os nomes das máquinas:

```
1.Inserir maquinas.
2.Inserir dados.
3.Apresentar dados das maquinas.
4.Guardar os dados num ficheiro binario.
5.Apresentar os dados num momento pedido pelo utilizador.
6.Contar as falhas das maquinas.
0.Sair do programa
Escolha a opcao que pretende: 1

Insira o numero de maquinas que pretende:2

Insira o nome da 1 maquina:
veg1

Insira o nome da 2 maquina:
veg2
```

Figura 6- Nomes das máquinas

Exemplo de output da função de inserir os dados da máquina:

```
Escolha a opcao que pretende: 2

Insira o nome da maquina onde quer colocar os dados: veg1

Insira a quantidade de Co2 da maquina: 10

Insira a quantidade de energia produzida pela maquina: 23

Insira a temperatura da maquina: 4

Insira a quantidade produzida pela maquina: 56

Insira a humidade da maquina atualmente: 32
```

Figura 7- Interface da função InsereDados

b) Mostrar todos os dados, de uma determinada máquina, guardados num intervalo de tempo indicado.

Esta função vai apresentar o nome de cada máquina e para cada máquina vai apresentar os seus dados através de printfs. A função e o output desta função são os seguintes:

```
void ApresentaDados(int count, nomesmaquinas ArrayNomes[200])
{
    plista aux = NULL;
    for(int i=0;i<count;i++)
    {
        printf("\nNome da maquina: %s", ArrayNomes[i].nome);
        aux = ArrayNomes[i].pprox;
        while(aux!=NULL)
        {
            printf("\nQuantidade de Co2 libertada pela maquina: %.1f", aux->Co2);
            printf("\nQuantidade de energia produzida pela maquina: %.1f", aux->energia);
            printf("\nQuantidade de humidade da maquina: %.1f", aux->humidade);
            printf("\nIndices de producao da maquina: %.1f", aux->producao);
            printf("\nTemperatura atual da maquina: %.1f", aux->temperatura);
            printf("\nData: %s", aux->data);
            aux = aux->prox;
        }
    }
}
```

Figura 8- Função para apresentar as máquinas e os seus dados

```
Escolha a opcao que pretende: 3

Nome da maquina: veg1
Quantidade de Co2 libertada pela maquina: 10.0
Quantidade de energia produzida pela maquina: 23.0
Quantidade de humidade da maquina: 32.0
Indices de producao da maquina: 56.0
Temperatura atual da maquina: 4.0
Data: Sat Jun 27 20:21:01 2020

Nome da maquina: veg2
```

Figura 9- Interface da função ApresentaDados

c) Guardar num ficheiro binário toda a *base de dados* para que não seja necessário inserir dados em cada nova execução do programa.

Nesta alínea, a função vai guardar os dados de cada máquina e o seu nome num ficheiro binário através de fwrites. Na figura abaixo podemos ver essa mesma função:

```
void GuardarDados(int count, nomesmaquinas ArrayNomes[200]){
    FILE * gravardados;
    plista aux = NULL;
    int n = 0;

    gravardados = fopen("dados.dat","wb");

    fwrite(&count,sizeof(int),1,gravardados);
    for(int i=0;i<count;i++){

        fwrite(&ArrayNomes[i].nome, sizeof(char)*20 , 1 , gravardados);
        n = ContaLista(ArrayNomes[i].pprox);
        fwrite(&n, sizeof(int), 1, gravardados);
        aux = ArrayNomes[i].pprox;
        while(aux != NULL){
            fwrite(aux, sizeof(dados) , 1 , gravardados);
            aux = aux->prox;
        }
    }
    fclose(gravardados);
}
```

Figura 10- Função para guardar os dados da máquina

d) Ler e colocar na estrutura, todos os dados guardados no ficheiro binário.

Esta função é semelhante à de guardar os dados, só que nesta em vez de guardar vai ler os dados que estão guardados no ficheiro binário através de freads. A função é a seguinte:

```
int CarregarDados(nomesmaquinas ArrayNomes[200])
{
    FILE * carregardados;
    carregardados = fopen("dados.dat", "rb");
    if(carregardados == NULL) {
        printf("\nFicheiro nao existe!");
        fclose(carregardados);
        return 0;
    }
    else {
        plista no;
        int AuxCount;
        int n = 0;
        fread(&AuxCount,sizeof(int),1,carregardados);
        for (int i = 0; i < AuxCount; i++) {

            fread(&ArrayNomes[i].nome, sizeof(char) * 20, 1, carregardados);
            fread(&n, sizeof(int), 1, carregardados);
            ArrayNomes[i].pprox = NULL;

            for (int j = 0; j < n; j++) {
                no = (plista) malloc(sizeof(dados));
                fread(no, sizeof(dados), 1, carregardados);
                no->prox = ArrayNomes[i].pprox;
                ArrayNomes[i].pprox = no;
            }
        }

        fclose(carregardados);
        return AuxCount;
```

Figura 11- Função para ler os dados do ficheiro binário

e) Retirar, da lista de uma máquina, os dados de um momento indicado pelo utilizador.

Nesta alínea, para o utilizador conseguir retirar os dados da lista de uma máquina no momento indicado pelo utilizador primeiro, pedimos, a este mesmo para inserir a data cujos dados pretende retirar (a data tem que ser exatamente igual a que esta quando se apresenta os dados). Se esta data for igual a uma das datas dos dados das máquinas então se essa máquina tiver pelo menos um no, o ponteiro paux vai guardar os dados do próximo nó dessa máquina e vai ser eliminado o primeiro no, ou seja, os primeiros dados e assim sucessivamente ate não haver mais dados dessa máquina que tenham essa data. Caso o paux não tenha nenhuns dados das máquinas então os dados da máquina vão ser igual aos próximos dados desta mesma máquina e é eliminado o aux, ou seja, elimina os primeiros dados da máquina. Por último, o paux vai ser igual ao aux, logo, já vai passar no segundo if do while se o aux for diferente de null e o aux passa a ser os próximos dados da máquina, para deste modo, andar com a lista para a frente. A função de retirar os dados da máquina de um dado momento é a seguinte:

```
void RetiraMomento(int count, nomesmaquinas ArrayNomes[200])
{
    char stringdata[200];

    printf("\nInsira o momento cujos dados pretende retirar (ex: Fri Jun 26 18:47:47 2020):");
    fflush(stdin);
    gets(stringdata);

    plista aux = NULL;
    plista paux = NULL;
    for(int i=0;i<count;i++)
    {
            aux = ArrayNomes[i].pprox;
            while(aux!=NULL)
            {
                if(strcmp(aux->data, stringdata) == 0)
                {
                    if(paux!=NULL)
                        paux->prox = aux->prox;
                    else {
                        ArrayNomes[i].pprox = aux->prox;
                    }
                    free(aux);
                }
                paux = aux;
                aux = aux->prox;
            }
    }
```

Figura 12- Função RetiraMomento

Exemplo de output desta função:

```
Nome da maquina: we
Quantidade de Co2 libertada pela maquina: 1.0
Quantidade de energia produzida pela maquina: 2.0
Quantidade de humidade da maquina: 5.0
Indices de producao da maquina: 4.0
Temperatura atual da maquina: 3.0
Data: Sun Jun 28 16:06:24 2020
```

Figura 13- Dados da máquina we

```
Escolha a opcao que pretende: 5

Insira o momento cujos dados pretende retirar (ex: Fri Jun 26 18:47:47 2020):Sun Jun 28 16:06:24 2020
```

Figura 14- Escolher a opção para eliminar os dados da máquina we

```
Escolha a opcao que pretende: 3

Nome da maquina: we
```

Figura 15- Dados eliminados

f) Contar, em cada máquina, os momentos em que ocorreram falhas (valores a 0).

Nesta função, para contar os momentos em que ocorrem falhas fizemos uma condição if em que se todas os valores da máquina tiverem a zero então a variável contador vai incrementar e depois é apresentado o número de zeros de cada máquina. A função é a seguinte:

```
void ContaFalhas(int count, nomesmaquinas ArrayNomes[200])
{
  int contador = 0;
  plista aux = NULL;
  for(int i=0;i<count;i++)
  {
    aux = ArrayNomes[i].pprox;
    while(aux!=NULL) {
      if (aux->energia == 0 && aux->producao == 0 && aux->humidade == 0 && aux->Co2 == 0 && aux->temperatura == 0) {
       strcpy(aux->data, "0");
       contador++;
            }
       aux = aux->prox;
       }
       printf("\nMaquina %s: ", ArrayNomes[i].nome);
       printf("\nNumero de erros desta maquina: %d", contador);
       contador = 0;
    }
}
```

Figura 16- Função para contar as falhas

g) Retirar de todas as listas os registos de falhas (valores a 0).

Esta alínea, é semelhante à função de retirar os dados de um dado momento, no entanto aqui só retira a lista se todos os valores da máquina forem zero. A função é a seguinte:

```
void RetiraFalhas(int count, nomesmaquinas ArrayNomes[200])
{
    plista aux = NULL;
    plista paux = NULL;
    for(int i=0;i<count;i++)
    {
        aux = ArrayNomes[i].pprox;
        while(aux!=NULL)
        {
            if(aux->energia == 0 && aux->producao == 0 && aux->humidade == 0 && aux->Co2 == 0 && aux->temperatura == 0 && (strcmp(a
            {
                if(paux!=NULL)
                    paux->prox = aux->prox;
                else {
                    ArrayNomes[i].pprox = aux->prox;
                }
                free(aux);
            }
            paux = aux;
            aux = aux->prox;
        }
    }
}
```

Figura 17- Função para retirar falhas

h) Procurar e listar dados que verificam dois critérios de pesquisa à escolha (por exemplo, data anterior a Junho de 2020 e temperatura superior a 40).

Nesta função, decidimos que só vai ser apresentado os dados das máquinas que tenham uma temperatura maior que 40 e uma produção maior que 200. A função e o output desta mesma são os seguintes:

```
void ProcuraCriterios(int count, nomesmaquinas ArrayNomes[200])
{
    plista aux = NULL;
    for(int i = 0;i<count;i++)
    {
        aux = ArrayNomes[i].pprox;
        while(aux!=NULL) {
          if (aux->temperatura > 40 && aux->producao > 200) {
           printf("\n\nMaquina com os criterios encontrada! Nome: %s", ArrayNomes[i].nome);
           printf("\nQuantidade de Co2 libertada pela maquina: %.1f", aux->Co2);
           printf("\nQuantidade de energia produzida pela maquina: %.1f", aux->energia);
           printf("\nQuantidade de humidade da maquina: %.1f", aux->humidade);
           printf("\nIndices de producao da maquina: %.1f", aux->producao);
           printf("\nTemperatura atual da maquina: %.1f", aux->temperatura);
           printf("\nData: %s", aux->data);
            }
            else
                printf("\nNao ha maquinas com esses criterios!");
            aux = aux->prox;
        }
    }
}
```

Figura 18- Função para procurar os dados que verificam dois critérios

```
Escolha a opcao que pretende: 8


Maquina com os criterios encontrada! Nome: re
Quantidade de Co2 libertada pela maquina: 10.0
Quantidade de energia produzida pela maquina: 3.0
Quantidade de humidade da maquina: 3.0
Indices de producao da maquina: 300.0
Temperatura atual da maquina: 50.0
Data: Sun Jun 28 16:39:40 2020
```

Figura 19- Interface da função ProcuraCriterios

i) Ordenar a lista de uma máquina por ordem crescente das datas.

Nesta função só depois de guardarmos os dados e voltarmos a executar o programa e listarmos os dados, é que aparecem as datas ordenadas (pelo menos é como nós estamos a interpretar), de cima para baixo. A função auxiliou de subprogramas anteriores para conseguirmos fazê-la, que são os seguintes:

```
//funções auxiliares para a alínea i
void DaValor(char *inicial, char *hora, char *dia, char *mes, char *ano) {

    char aux[30];
    strcpy(aux, inicial);
    strtok(aux, " ");//DS
    strcpy(mes, strtok(NULL, " "));
    printf("mes: %s\n",mes);//c
    strcpy(dia, strtok(NULL, " "));
    printf("dia: %s\n",dia);//c
    strcpy(hora, strtok(NULL, " "));
    printf("hora: %s\n",hora);//c
    strcpy(ano, strtok(NULL, " "));
    printf("ano: %s\n",ano);//c
}
```

Figura 20- Função DaValor

Esta de cima serve para separar a data e guardar os “pedaços” separados em diferentes variáveis.

A seguinte serve para inverter a ordem das listas:

```
void Switch(plista * lst1, plista * lst2)//c: coloquei *
{
    plista aux = (plista)malloc(sizeof(dados));

    strcpy(aux->data, (*lst1)->data);
    aux->producao = (*lst1)->producao;
    aux->energia = (*lst1)->energia;
    aux->Co2 = (*lst1)->Co2;
    aux->humidade = (*lst1)->humidade;
    aux->temperatura = (*lst1)->temperatura;
    aux->prox=(*lst1)->prox;

    strcpy((*lst1)->data, (*lst2)->data);
    (*lst1)->producao = (*lst2)->producao;
    (*lst1)->energia = (*lst2)->energia;
    (*lst1)->Co2 = (*lst2)->Co2;
    (*lst1)->humidade = (*lst2)->humidade;
    (*lst1)->temperatura = (*lst2)->temperatura;
    (*lst1)->prox=(*lst2)->prox;

    strcpy((*lst2)->data, aux->data);
    (*lst2)->producao = aux->producao;
    (*lst2)->energia = aux->energia;
    (*lst2)->Co2 = aux->Co2;
    (*lst2)->humidade = aux->humidade;
    (*lst2)->temperatura= aux->temperatura;
    (*lst2)->prox=aux->prox;
```

Figura 21-Função Switch

Por fim, temos o algoritmo de ordenação:

```
void OrdenaDataCrescente(int count, nomesmaquinas ArrayNomes[200])
{
    plista aux = NULL, aux1 = NULL;
    char hora1[15], hora2[15], dia1[6], mes1[6], ano1[6], dia2[6], mes2[6], ano2[6];
    for(int i=0;i<count;i++)
    {
        aux = ArrayNomes[i].pprox;
        aux1=aux->prox;
        while(aux != NULL && aux1!= NULL)
        {
            DaValor(aux->data, hora1, dia1, mes1, ano1);
            printf("data: %s\nhora: %s\ndia: %s\nmes:%s\nano: %s\n",aux->data,hora1,dia1,mes1,ano1);//c
            DaValor(aux1->data, hora2, dia2, mes2, ano2);
            printf("data: %s\nhora: %s\ndia: %s\nmes:%s\nano: %s\n",aux1->data,hora2,dia2,mes2,ano2);//c

            if(strcmp(ano1, ano2)>0)
                Switch(&aux, &aux1);//c:&
            else
            if(strcmp(ano1, ano2) == 0)
            {
                if(strcmp(mes1, mes2) > 0)
                    Switch(&aux, &aux1);//c:&
            }
```

Figura 22- Função OrdenaDataCrescente

```
            else
            if(strcmp(ano1, ano2) == 0 && strcmp(mes1, mes2) == 0)
            {
                if(strcmp(dia1, dia2)>0)
                    Switch(&aux,&aux1);//c:&
                else if(strcmp(hora1, hora2)>0)
                    Switch(&aux, &aux1);//c:&
            }
            aux = aux->prox;
            aux1 = aux1->prox;

        }
    }
}
```

Figura 23- Continuação

Output após guardar os dados e executar de novo o programa:

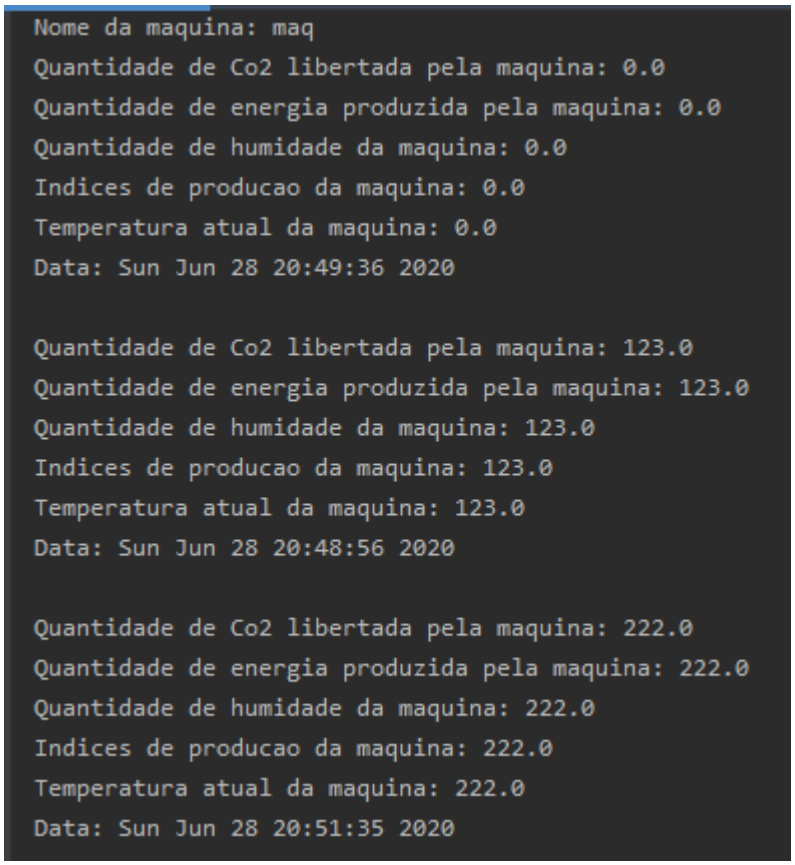

Figura 24- Listas ordenadas por ordem crescente

j) Criar um ficheiro csv com os dados de cada máquina

Nesta última função, para guardar os dados em ficheiro csv é semelhante a guardar em ficheiro texto só que aqui o nome do ficheiro em vez de ser .txt é .csv e os valores tem que ser separados por (;). A função é a seguinte:

```
int GravaCsv(int count, nomesmaquinas ArrayNomes[200])
{
    FILE *f = fopen("data.csv", "w");
    plista aux = NULL;
    if(f == NULL)
        return -1;
    else
        for(int i=0;i<count;i++)
        {
            aux = ArrayNomes[i].pprox;

            fprintf(f, "\nMaquina:; ");
            fprintf(f, "%s;", ArrayNomes[i].nome);
            fprintf(f, "\n\n");
                while (aux != NULL)
                {
                    fprintf(f, "Data/Hora;Temperatura;Humidade;CO2 emitido;Energia;Producao;");
                    fprintf(f, "\n");
                    fprintf(f, "%s;%.1f;%.1f;%.1f;%.1f;%.1f", aux->data, aux->temperatura, aux->humidade, aux->Co2, aux->energia,
                    aux = aux->prox;
                }
        }
    fclose(f);
}
```

Figura 25- Função para guardar os dados num ficheiro csv

3 Conclusão

Com este último trabalho, familiarizamo-nos melhor com as listas ligadas e ficamos a ter uma maior noção sobre a matéria das listas e das estruturas também. Assim, acho que de um modo geral foi um trabalho bem conseguido e que nos deu mais bases para no futuro sabermos lidar com listas ligadas.