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

Licenciatura Tecnologias e Sistemas de Informação

Programação II

2019/20

Relatório

Ficha 7

Jorge Miguel dos Santos Martins Nº a2019100813

16/06/2020

Índice

1 Introdução

Sendo esta a última ficha, esta é um resumo de toda a matéria dada neste semestre, por isso podemos confirmar que é um ótimo material de estudo para a manipulação de ficheiros através de estruturas com listas ligadas. Existiram alguns obstáculos na construção do código para que este funcionasse corretamente, pois em teoria estaria com uma arquitetura correta, mas nem sempre se comporta como esperamos que se comporte, pois existem alguns pormenores que nos escapam.
Seguindo a mesma logica neste relatório irei focar nos aspetos mais importantes a relatar, decisões tomadas, pormenores do meu código que acho importante realçar para que a leitura seja mais simples. Deixei também vários comentários no meu código a explicar a ação que estava a tomar, alguns já são retirados do código realizado da ficha 5 exercício 2 e para aqui adaptados.

2 Trabalho realizado

Neste trabalho estão dois exercícios inseridos num apenas pensei que assim fosse mais logico visto que um está inserido no outro.
Apresentando as minhas estruturas, uma pensada para as listas ligadas que são os dados relativos a cada máquina e outra para que seja inserida num vetor de estruturas que consiste para que seja inserida cada máquina.

```
typedef struct data_maquinas * point; //estrutura dos dados de cada maquina
typedef struct data_maquinas {
char data[40];
float temp;
float humi;
float co2;
float EnerC;
float producao;
point prox;

} dados;


typedef struct estrutura_maquinas { //estrutura que tem os apontadores para cada maquina**
  char nome[100]; //nome da maquina
  point head; //vai apontar para cada inicio de lista de uma maquina
} maquinas;
```

Figura 1 - Estruturas do programa

A primeira é a estrutura que vai ser utilizada nas listas ligadas enquanto a segunda é a estrutura que vai ser utilizada como vetor de estruturas ao qual tem dois elementos, um sendo o nome e outro sendo o apontador para o início da lista.
Na figura 2 que se encontra abaixo, está um dos subprogramas de inserção de valores na lista ligada das máquinas, aqui o que tenho a referenciar em relação ao que já foi usado em fichas anteriores é o facto de eu em vez de utilizar a função “ctime” da biblioteca “time.h”, que já tínhamos usado na ficha 2 deste semestre, escolhi utilizar uma “struct tm” diretamente (sim porque a ctime também utiliza mas num modo de escrita especifico) porque assim consegui utilizar a data toda de forma numérica em vez de conter alguns nomes acerca dos meses e dias da semana. Isto assim a estrutura da minha data mesmo sendo passada para string ficará “dd/mm/yyyy – hh:mm”, todos estes valores ficam em numerário.
Em vez de utilizar o “ctime”, utilizo a função “localtime” da mesma biblioteca e em seguida ao passar para string a estrutura utilizo a função “strftime” ao qual o modo desta função é a variável que vai receber os valores, entre parenteses o tipo de valores que quero, sendo o “%d” o número de dias de 1 a 31, “%m” o mês de 1 a 12, “%Y” (em grande) para que o valor seja o ano completo ex 2020, em seguida temos os valores que serão para as horas sendo o “%H” e “%M” ambos em grande para que as horas sejam de 0 a 23 e minutos de 0 a 59. Tudo isto foi retirado do site “ http://www.cplusplus.com/reference/ctime/strftime/ “.

```c
void junta_no_ini_lista(point * lst) { //inserir dados com data automatica..
  struct tm *tempo;
  time_t dat;
  float valor;
  char aux[100];
  time(&dat);
  point no = (point)malloc(sizeof(dados));
  if (no == NULL){
    perror("ERRO!!! Nao ha memoria disponivel...");
    exit(-1);
  }
  else{
   puts("Insira temperatura"); scanf("%f", &valor); flush_in();
   no->temp=valor;
   puts("Insira humidade"); scanf("%f", &valor); flush_in();
   no->humi=valor;
   puts("Insira co2"); scanf("%f", &valor); flush_in();
   no->co2=valor;
   puts("Insira energia consumida"); scanf("%f", &valor); flush_in();
   no->EnerC=valor;
   puts("Insira produção da maquina"); scanf("%f", &valor); flush_in();
   no->producao=valor;
   tempo = localtime(&dat); //uso na mesma o localtime e a seguir retiro a info dele para string com o strftime
   strftime(no->data,40,"%d/%m/%Y - %H:%M", tempo); //d para dia m para mes e Y em grande para que o ano seja completo
   no->prox = * lst; // o ponteiro desta estrutura fica com valor null (ficando a ser o ultimo da lista)
    * lst = no; //o valor apontado por lst passa a ser o valor que o no inicializou a lista
  }
}
```

Figura 2 - Exemplo de inserir valores

Seguidamente passo a explicar como foi realizado no caso da inserção de valores manuais, pois também uso a função “localtime”, mas utilizo a função “mktime” para passar o valor dentro da estrutura seja passado para a forma “time_t”.
Desde já referenciar que em vez de utilizar um ponteiro global da “struct tm”, para não criar confusões na criação dos dados pois não tenho completo domínio sobre esta tipologia, decidi a iniciar com nomes diferentes dentro de cada subprograma que utilizo este tipo de “struct”. Podemos observar que no subprograma que os dados são inseridos e a hora é automaticamente quase em tempo real tem o nome de tempo, mas na figura 3 que já demonstra uma inserção de dados manuais o nome que dei é juntaInicio, ambos os ponteiros fazem o mesmo, mas apenas os diferenciei nos nomes.
Olhando para o código utilizado no subprograma da figura 3 podemos observar que decidi pedir os elementos de forma separada no que consta da data, basicamente as variáveis auxiliares inteiras que vão receber esses valores, serão usadas para dar à estrutura “struct tm” os valores de forma manual e seguidamente para que os valores estabilizem entre todos os elementos dessa estrutura (para que não tenha que pedir muita informação ao utilizador) uso a função “mktime”. Podemos ver que seguido do uso da função “mktime”, utilizo como no subprograma “automático” a função “strftime”, para que seja passada a data para string.

```c
void junta_no_ini_listaManual(point * lst) { //INSERÇÃO MANUAL DOS DADOS
  struct tm *juntaInicio;
  int ano, mes, dia, hora, min, seg, val;
  time_t dat;
  time (&dat);
  float valor;
  char aux[100];
  point no = (point)malloc(sizeof(dados));
  if (no == NULL){
    perror("ERRO!!! Nao ha memoria disponivel...");
    exit(-1);
  }
  else{
   puts("Insira temperatura"); scanf("%f", &valor); flush_in();
   no->temp=valor;
   puts("Insira humidade"); scanf("%f", &valor); flush_in();
   no->humi=valor;
   puts("Insira co2"); scanf("%f", &valor); flush_in();
   no->co2=valor;
   puts("Insira energia consumida"); scanf("%f", &valor); flush_in();
   no->EnerC=valor;
   puts("Insira produção da maquina"); scanf("%f", &valor); flush_in();
   no->producao=valor;
   juntaInicio = localtime (&dat);
   puts("Insira dia"); scanf("%d", &dia); flush_in();
   puts("Insira mes"); scanf("%d", &mes); flush_in();
   puts("Insira ano"); scanf("%d", &ano); flush_in();
   puts("Insira hora - Apenas a hora não os minutos"); scanf("%d", &hora); flush_in();
   puts("Insira minutos"); scanf("%d", &min); flush_in();
   puts("Insira segundos"); scanf("%d", &seg); flush_in();
   if(dia ==0 && mes == 0 && ano == 0 && hora == 0 & min == 0 && seg == 0){ //por causa das falhas...
     strcpy(no->data, "0");
     no->prox = * lst; // o ponteiro desta estrutura fica com valor null (ficando a ser o ultimo da lista)
      * lst = no; //o valor apontado por lst passa a ser o valor que o no inicializou a lista
   }
   else{
     juntaInicio->tm_year = (ano - 1900);
     juntaInicio->tm_mon  = (mes -1);
     juntaInicio->tm_mday = dia;
     juntaInicio->tm_hour = hora;
     juntaInicio->tm_min = min;
     juntaInicio->tm_sec = seg;
     val=mktime(juntaInicio); //função que tranforma em time_t, retira os dados existentes e adapta-os para o tempo..
     if( val == -1 ) {
        printf("Erro: impossivel o mktime ter sido usado\n");
        return;
     }
     else{
       strftime(no->data,40,"%d/%m/%Y - %H:%M", juntaInicio); //após ter os dados transformo-os novamente em string
       no->prox = * lst; // o ponteiro desta estrutura fica com valor null (ficando a ser o ultimo da lista)
        * lst = no; //o valor apontado por lst passa a ser o valor que o no inicializou a lista
    }
  }
 }
}
```

Figura 3 - Exemplo inserir manualmente

Ao observar o código podemos ver que igualo o ano e o mês ao que foi inserido menos 1900 e menos um, pois a tabela guarda valores dessas diferenças e assim ao gerar data é com esses valores que são adicionados e subtraído para dar a data correta.

```c
int comparaDatas(char data[40], int dia1, int mes1, int ano1, int dia2, int mes2, int ano2){
  char dataDia[3], dataMes[3], dataAno[5];
  int j=0, t=0, dia, mes, ano;
  for(int i=0; i<10; ++i){ //ciclo que vai retirar caracteres da string data
    if(i>=0 && i<2){
      dataDia[i]=data[i];
    }
    else{
      if(i>2 && i<=4){
        dataMes[j]=data[i];
        j++;
      }
      else{
        if(i>5 && i<=9){
          dataAno[t]=data[i];
          t++;
        }
      }
    }
  }
  dataDia[2]='\0'; //passar de vetor de caracteres para string
  dataMes[2]='\0';
  dataAno[4]='\0';

  dia = strtol(dataDia, NULL, 10); //transforma uma string em um inteiro de base 10
  mes = strtol(dataMes, NULL, 10);
  ano = strtol(dataAno, NULL, 10);
```

Figura 4 - Compara Datas

Dentro do subprograma de comparar datas está um ponto interessante a realçar do meu código. Após o utilizador indicar o intervalo de datas que quer que seja filtrado numa das máquinas, é chamada um subprograma de comparação de datas, pois irei comparar a data numérica de uma string com os valores inteiros inseridos por um utilizador.
O ciclo inicial for serve para estrategicamente retirar os caracteres acerca do dia, mês e ano que a string da data contem e assim em seguida a transformar em inteiro, através da função “strtol” que transforma em um número de base 10. Em seguida e não visível nesta figura é que são feitas as comparações entre anos, meses e dias para descobrir se está ou não dentro do intervalo de tempo desejado.

```c
void saveDataBase(maquinas vetor[TAMANHO], int counter){
  FILE *saveBin;
  int lista=0, i=0;
  point atual;
  saveBin=fopen("basedadosMaquinas.dat", "wb");


  if(saveBin==NULL){
    printf("Erro ao abrir ficheiro, clique para sair\n"); getchar();
    fclose(saveBin);
    return;
  }
  else{
    fwrite(&counter, sizeof(int), 1, saveBin);
    fwrite(vetor, sizeof(maquinas)*10, 1, saveBin);
    while(i<counter){
      atual=vetor[i].head;
      lista=contaListas(vetor[i].head);
      fwrite(&lista, sizeof(int), 1, saveBin); //gravar o numero de listas que estão inseridas / contruídas..
      if(lista > 0){
        while(atual != NULL){ //preferi guardar elemento a elemento assim tenho noção que foi tudo gravado.
          fwrite(&(atual->data), sizeof(char)*40, 1, saveBin);
          fwrite(&(atual->temp), sizeof(float), 1, saveBin);
          fwrite(&(atual->humi), sizeof(float), 1, saveBin);
          fwrite(&(atual->co2), sizeof(float), 1, saveBin);
          fwrite(&(atual->EnerC), sizeof(float), 1, saveBin);
          fwrite(&(atual->producao), sizeof(float), 1, saveBin);
          atual=atual->prox;
        }
        i++;
      }
      else{ //o outro i esta dentro do if, isto porque se a maquina não tiver dados o programa rebenta ao gravar
        i++;
      }
    }
  }
  fclose(saveBin);
}
```

Figura 5 - Gravação de Dados

Através da figura 5, podemos perceber que para a gravação de dados num ficheiro binário optei pela gravação de elemento a elemento, retirando apenas a inicial gravação de todo o vetor num bloco e em seguida de forma linear gravar os dados de máquina em máquina e lista em lista dentro de cada máquina.
A opção do counter e da variável lista que vai buscar o numero de nós que existem dentro de cada lista, serve para que depois ao ler os dados do ficheiro binário seja possível calibrar todos os dados para os seus devidos locais. O Counter indicará quantas máquinas estão inseridas e assim não se passar do tamanho máximo do vetor e a variável lista para que ao se ler os dados se saiba quantas lista irão ser lidas.
Caso a lista seja 0 a máquina é gravada na mesma dentro do ficheiro binário.

```
int loadDataBase(maquinas vetor[TAMANHO], int counter){ //subprograma que vai fazer a leitura do que está no ficheiro..
 FILE *loadBin;
 int lista=0, i=0;
 point novo, no, atual;

 loadBin=fopen("basedadosMaquinas.dat", "rb");

 if(loadBin==NULL){
   printf("Erro ao abrir ficheiro ou não existente, clique para continuar\n"); getchar();
   fclose(loadBin);
   return 0;
 }
 else{
   fread(&counter, sizeof(int), 1, loadBin);//controle de não sair do vetor o numero de maquinas..
   fread(vetor, sizeof(maquinas)*10, 1, loadBin);
   while(i<counter){
     fread(&lista, sizeof(int), 1, loadBin); //para ir buscar o tamanho das listas que foram gravadas nos dados..

     vetor[i].head=crialistamaquinas(); //deixar o ponteiro a null, para se não houver dados dar lista vazia
     atual=vetor[i].head; //o mesmo que estar a null
     if(lista>0){ //para por no inicio da lista
       no = (point)malloc(sizeof(dados));
       fread(&(no->data), sizeof(char)*40, 1, loadBin);
       fread(&(no->temp), sizeof(float), 1, loadBin);
       fread(&(no->humi), sizeof(float), 1, loadBin);
       fread(&(no->co2), sizeof(float), 1, loadBin);
       fread(&(no->EnerC), sizeof(float), 1, loadBin);
       fread(&(no->producao), sizeof(float), 1, loadBin);

       no->prox = vetor[i].head; //o mesmo que estar igual a NULL
       vetor[i].head = no;
       atual = vetor[i].head; //fica a apontar para o mesmo local que o head..
     }
     lista=(lista-1);
     while(lista > 0){ //caso seja 0 inicialmente -1 não é maior que 0

       novo = (point)malloc(sizeof(dados));
       fread(&(novo->data), sizeof(char)*40, 1, loadBin);
```

Figura 6 - Leitura dos Dados

Na leitura dos dados é necessário seguir estritamente como se foi gravados para que os dados não caiam em sítio errado. O counter é lido em primeiro pois foi o primeiro a ser gravado, em seguida o vetor, e então é que irá entrar no ciclo de leitura para os dados das máquinas. Dentro desse ciclo é lida a lista (tamanho da lista ligada), e logo de seguida é inicializado o ponteiro “head” como NULL. Esta opção está antes de entrar na leitura dos dados para que se a lista for 0 o ponteiro é inicializado corretamente e assim o programa não rebente ou o ponteiro fique a apontar para um local aleatório de memoria não recomendável. No seguimento é feita a primeira leitura dos dados e inseridos no primeiro lugar dando assim o ponteiro head a indicação que fique a apontar para o inicio da lista, o que no ciclo while a seguir já não será o head (ponteiro inicial da lista) que ficará apontar mas sim aquele que está em ultimo lugar (o que está em NULL indicando o fim).

Para o subprograma “retiraFalhas” usei a mesma logica que foi usada na ficha 5 no exercício 2 ao qual nos pedia para remover uma ocorrência na lista, visto serem todos os valores a 0, é então que corre a lista e ao encontrar uma lista com falhas, conecta a lista anterior à próxima a essa retirando-a assim da lista ligada.

```
void retiraFalhas(point *lst){
  int v, datt;
  point ant, atual;
  if((v=listavaziamaquinas(*lst))==0){
    atual=*lst;
    datt=strtol(atual->data, NULL, 10);
    while(datt == 0 && atual->temp == 0 && atual->humi == 0 && atual->co2 == 0 && atual->EnerC == 0 && atual->producao == 0 && atual != NULL){
      atual=atual->prox;
      *lst=atual;
      if(*lst == NULL){ // caso toda a lista tenha valores iguais adaptei apra que fizesse primeiro um ciclo em vez de apenas uma condição
        break;
      }
      datt=strtol(atual->data, NULL, 10); //strtol apenas vai encontrar o 0
    }
    if((*lst)->prox == NULL){
      puts("Terminado");
    }
    else{
      while(atual != NULL){
        ant=atual;
        atual=atual->prox;
        datt=strtol(atual->data, NULL, 10);
        if(datt == 0 && atual->temp == 0 && atual->humi == 0 && atual->co2 == 0 && atual->EnerC == 0 && atual->producao == 0){ // caso seja igual é só ligar o an
          ant->prox = atual->prox;
          atual=atual->prox;
        }
      }
      puts("Terminado");
    }
}
  else
    puts("Lista encontra-se vazia");
```

Figura 7 - Retira Falhas

De também realçar o facto que o algoritmo primeiro verifica se as falhas estão logo no primeiro lugar da lista pois o comportamento de troca de ponteiros é diferente.
Nos subprogramas seguintes não existe nada que seja de valor realçar, pois uso o mesmo algoritmo para ordenar o vetor e de trocar os valores que já usei anteriormente nas outras fichas e na comparação de datas uso a mesma técnica que já aqui referenciei, apenas só mais pormenorizado na comparação até à hora. Para a verificação de um filtro com dois parâmetros escolhi a data ser superior (reutilizo o subprograma de comparar as datas até a sua hora) e comparo diretamente com o valor da produção dado pelo utilizador com o que se encontra em cada lista.
Para o subprograma de gravação em ficheiro de excel, seguindo as instruções da Docente e gravando como se de um ficheiro de texto se tratasse não foi preciso decidir muito como realizar as tarefas pretendidas.

```c
void gravarExcel(maquinas vetor[TAMANHO], int counter){ //subprograma para gravar em excel como a stora ensina na ficha
  FILE *saveEx;
  int i=0;
  point atual;
  saveEx=fopen("basedadosMaquinas.csv", "w");


  if(saveEx==NULL){
    printf("Erro ao abrir ficheiro, clique para sair\n"); getchar();
    fclose(saveEx);
    return;
  }
  else{
    while(i<counter){
      fprintf(saveEx, "\n %s ;", "Maquina: ");
      fprintf(saveEx, "%s \n\n", vetor[i].nome);
      atual=vetor[i].head;
      fprintf(saveEx, "%s ; %s ; %s ; %s ; %s ; %s \n", "Data", "temperatura", "Humidade", "Co2", "Energia Consumida", "Producao");
      while(atual != NULL){
        fprintf(saveEx, "%s ;", atual->data);
        fprintf(saveEx, "%.2f ;", atual->temp);
        fprintf(saveEx, "%.2f ;", atual->humi);
        fprintf(saveEx, "%.2f ;", atual->co2);
        fprintf(saveEx, "%.2f ;", atual->EnerC);
        fprintf(saveEx, "%.2f \n", atual->producao);
        atual=atual->prox;
      }
      i++;
    }
  }
  fclose(saveEx);
}
```

Figura 8 - Gravação em Excel

3 Conclusão

Esta foi uma ficha de trabalho que muita pesquisa tive de realizar devido ao facto de tentar utilizar algumas ferramentas por mim ainda não dominadas, o que me fez ocupar grande parte do meu tempo de trabalho na leitura acerca de documentação da gravação de dados em ficheiros binários de diferentes formas e como se comportam as listas ligadas nessas gravações.

Em vários momentos desta ficha reutilizei algoritmos já feitos por mim em outras fichas, o que foi simplesmente necessário adaptar o código. Sei da existência de algumas variáveis apêndice nesta minha ficha, mas devido ao facto de o meu atom não as detetar e de ter reestruturado o meu código penso que tenha perdido a noção de onde algumas se possam encontrar, ou outras que penso que estejam a ser utilizadas mas a IDE indica que não (ex da variável v que utilizo a verificar as máquinas).

Para as referências irei deixar toda a documentação que acedi para conseguir perceber como arquitetar o meu código.

4 Referências

Listar consultas usadas

- Luis Damas Liguagem C
- http://www.cplusplus.com/reference/ctime/strftime/
- https://www.tutorialspoint.com/c_standard_library/c_function_strftime.htm
- https://pt.stackoverflow.com/questions/272801/como-converter-string-para-o-tipo-int-em-c
- https://pt.stackoverflow.com/questions/126793/como-declarar-uma-vari%C3%A1vel-de-data-em-c
- https://stackoverflow.com/questions/13658756/example-of-tm-use
- https://pubs.opengroup.org/onlinepubs/009695399/functions/strptime.html
- https://edisciplinas.usp.br/pluginfile.php/3463219/mod_resource/content/0/Aula%2011.pdf
- http://www.cplusplus.com/reference/ctime/mktime/
- https://stackoverflow.com/questions/39673816/want-to-perform-date-time-value-manipulation-using-struct-tm
- https://stackoverflow.com/questions/15555406/how-to-compare-two-time-stamp-in-format-month-date-hhmmss-to-check-ve-or-v
- https://stackoverflow.com/questions/5950497/comparing-dates-in-c-with-using-time-h-library
- https://stackoverflow.com/questions/5378778/what-does-d-xopen-source-do-mean
- Trabalhar com tempo! - António Manuel de Sousa Barros Departamento de Eng. Informática ISEP – Instituto Superior de Engenharia do Porto Maio de 2003

5 Anexos

Exer1-2FINAL.c