Documentação

Igor Guilherme Bianchi 558400

Exercício B

No primeiro exercício resolvido foi utilizado o algoritmo força bruta, os testes foram feitos digitando o nome do arquivo no .exe e não por entrada padrão. O código primeiramente passa todos os valores contidos no arquivo para uma struct INFO e é adicionado mais um campo, carrasco_optimus, que é a divisão do índice do carrasco mamata pela duração da aula, isso facilita na hora da comparação para verificar qual horário compensa mais. É criado também um vetor auxiliar, do mesmo tipo INFO, para guardar os valores que serão utilizados no resultado. A primeira comparação é em relação ao dia, se for conflitante o algoritmo segue para a próxima comparação, se não marca a variável inválida como falsa.

```
if(strcmp(info[i].dia, saida[j].dia) != 0)
    invalido = false;
```

Caso o dia for conflitante, a próxima comparação verifica se as horas estão contidas no intervalo verificado. Se não estiverem a posição recebe inválido = falso, caso contrário vai para a próxima comparação.

```
else if(info[i].hora2 <= saida[j].hora1 || info[i].hora1 >= saida[j].hora2)
        invalido = false;
```

Agora é comparado o valor do carrasco_optimus, se for maior que o comparado, o valor da posição do resultado é trocado pelo da posição do vetor original. Caso contrário vai para última verificação. O break ocorre porque a posição já foi trocada, então não são necessárias mais verificações. Inválido recebe true pelo mesmo motivo de chamar break, para não ser alocado essa posição no vetor resultado sendo que ela já foi trocada.

```
else if(info[i].carrasco_optimus > saida[j].carrasco_optimus){
    invalido = true;
    saida[j] = info[i];
    info[i].escolhido = true;
    break;
}
```

Para finalizar essa primeira parte, caso nenhum dos casos acima sejam atingidos, significa que o valor é igual ou pertence ao intervalo de horários mas o seu índice é menor que todos os verificados, então recebe invalido = true e é chamado break, pois já existe um índice maior nesse intervalo de horário.

```
else{
        invalido = true;
        break;
}
```

Caso no final de todas as comparações a variável invalido fique como falsa, o valor verificado é passado para o vetor de resultado, em uma nova posição.

```
if(!invalido){
        saida[aux] = info[i];
        aux++;
        invalido = true;
        info[i].escolhido= true;
}
```

Pode ocorrer valores iguais no vetor resultado, então é feita uma nova verificação para deixar como índice de carrasco = 0, o que não interfere na soma final para o maior índice possível.

```
for(j = 0; j < aux; j++){
        for(i = j+1; i < aux; i++){
                if(strcmp(saida[i].dia, saida[j].dia) == 0 && (saida[j].hora1 == saida[i].hora1 && saida[j].hora2 == saida[i].hora2)){
                        if(saida[j].carrasco_optimus <= saida[i].carrasco_optimus)
                                saida[j].carrasco = 0;
                }
        }
}
```

Por fim é calculado o valor máximo utilizando o vetor resultado.

```
for(j = 0; j < aux; j++)
        k += saida[j].carrasco;
```

Nos testes realizados, cerca de 60% das entradas disponibilizadas no moodle tiveram resultados iguais ao que era esperado. Em relação aos 40% de saídas não esperadas, elas não tiveram um erro grande, porém não foi possível determinar o caso que estava faltando para ser verificado no código para que estas saídas se adequassem ao esperado.