INSTITUTO POLITÉCNICO DE lEIRIA

Engenharia Informática – Programação Avançada

2.º Ano – 1.º Semestre

PROJETO PRÁTICO

---

paCodec Tool – Etapa 1

Paulo Penicheiro nº2130628 e por Ruben Bernardo 2130664 declaram sob compromisso de honra que o presente trabalho (código, relatórios e afins) foi integralmente realizado por nós, sendo que as contribuições externas se encontra claramente e inequivocamente identificadas no relatório e no próprio código fonte. Mais se declara que os estudantes acima identificados não disponibilizaram o código ou partes dele a terceiros.

PAULO PENICHEIRO

N 2130628

ENHENHARIA INFORMATICA

Prof. PATRÍCIO DOMINGUES

LEIRIA, NOVEMBRO 2015

RUBEN BERNARDO

N º 2130664

ÍNDICE

1 - Introdução 1
2 - Funções Implementadas 1
Funcionalidades comuns às funções – ficheiro utils.c 1
Função about 1
Função calculatePSNR 1
Função install_signal_handler 2
Função process_signal 2
Função decode_pgm 2
Função deode_dir 2
Função validate_extension 3
Funcionalidades de gestão de ficheiros – ficheiro filehandler.c 3
Função skip_comments_spaces 3
Função read_cod_file 3
Função read_cod_header 3
Função read_dictionary 4
Função read_header_dict 4
Função load_cod_file_to_struct 4
Função alocate_dict_blocks 4
Função load_blocks_to_struct 4
Função dealloc_dict_blocks 4
Função read_file 4
Função allocate_matrix 5
Função dealloc_matrix 5
Função read_header 5
Função load_matrix_to_struct_p2 5
Função load_matrix_to_struct_p5 5
Função write_pgm_file 5
Funções Principais – ficheiro main.c 6
Função main 6
3 - Conclusão 6
4 - Objetivos 7
Objetivos Atingidos 7
Objetivos Não Atingidos 7
5 - Bibliografia 7

1 - Introdução

O relatório descreve as atividades realizadas no âmbito da execução do trabalho prático do curso de Engenharia Informática na disciplina de Programação Avançada – 1.ª Etapa.

Consiste, na implementação de um utilitário em modo consola chamado paCodec cujo objetivo é permitir efetuar a compressão/descompressão com perda, de imagens

Nas páginas seguintes é possível encontrar as opções tomadas na criação do programa, os objetivos cumpridos e não cumpridos nesta 1ª Etapa do trabalho.

2 - Funções Implementadas

Funcionalidades comuns às funções – ficheiro utils.c

Adotámos para as funções do utilitário paCodec que são mais generalistas e comuns às funções principais, de as incluir no ficheiro *utils.c,* estando os respetivos protótipos declarados no ficheiro *utils.h.*
Passamos a descrever essas mesmas funções:

Função about

Esta função é desencadeada caso seja dado o argumento *--about* (forma longa) ou *–A* (forma abreviada), e mostra a informação dos alunos que desenvolveram o utilitário paCodec.

Função calculatePSNR

Esta função tem com objetivo calcular a qualidade da compressão/descompressão, visto a codificação estar assente no método de padrões, logo com perdas, assim recorrendo a uma métrica dada (*PSNT – Peak Signal-To-Noise Ratio*).
A função é desencadeada caso seja dado o argumento *--PSNR*(forma longa) ou *–P* (forma abreviada).
Em traços gerais esta funcionalidade recebe num só argumento o ficheiro original e o ficheiro descodificado, separados por vírgula, de seguida recorrendo à funçáo strtok() trata-

se o argumento de forma a separar os ficheiros da vírgula e poder carregar os ficheiros para a suas respetivas estruturas e podermos aceder aos seus dados.
Após os ficheiros carregados nas suas estruturas procede-se ao cálculo da fórmula dada pelo PSNR, apresentando no final o resultado da operação assim como o tempo despendido com a mesma.

Função install_signal_handler

Esta função tem como finalidade analisar um sinal de sistema ativado pelo utilizador. No caso desta primeira etapa apenas será avaliado o sinal SIGINT e caso haja alguma interrupção da aplicação o sistema trata de processar a interrupção através da função process_signal.

Função process_signal

Esta função tem como objetivo processar o sinal ativado pelo utilizador e agir em conformidade fechando também todos os ficheiros abertos e informando o utilizador que a aplicação foi terminada.

Função decode_pgm

Esta função é desencadeada caso seja dado o argumento *--decode* (forma longa) ou *–d* (forma abreviada) e é responsável pela descodificação de ficheiros de imagem do tipo .cod transformando-as num ficheiro do tipo .pgm, com perdas de qualidade.
A função recebe a estrutura do ficheiro .cod e a estrutura do respetivo dicionário, mais o nome do ficheiro codificado assim como o nome do respetivo dicionário.
Como algoritmo de descodificação, analisa cada valor do ficheiro .cod, atribuindo à matriz o valor de acordo com o índice do dicionário, preenchendo também o cabeçalho da estrutura do ficheiro descodificado.

Função deode_dir

Esta função é desencadeada caso seja dado o argumento *–decode-dir* (forma longa) ou *–R* (forma abreviada) e é responsável pela descodificação de todos os ficheiros do tipo.cod, existentes no directório passado como argumento [Não totalmente implementada]

Função validate_extension

Função responsável pela validação da extensão do ficheiro dado como parâmetro de entrada recebendo ainda o tipo de extensão a validar.

De modo a validar a extensão do ficheiro dado como parâmetro de entrada recore à função *strrchr()* que obtém o conteúdo presente numa string que se situa após a verificação da primeira ocorrência de um determinado caracter. Esta função vai pesquisar o caracter "." a partir do fim do nome do ficheiro e de seguida comparar a string obtida com a string de extensão a validar, devolvendo "1" caso a função strcmp detete que são iguais e "0" caso contrário

Funcionalidades de gestão de ficheiros – ficheiro filehandler.c

Função skip_comments_spaces

Esta função é responsável pela identificação de comentários no parsing de toso os ficheiros necessários, sendo que sempre que encontra o caracter '#' no início da linha, passa automaticamente para a leitura da próxima linha até ao final do ficheiro, caso contrário processa a leitura da linha corrente.

Esta função é recursiva chamando-se a si própria

Função read_cod_file

Esta função faz a leitura do ficheiro de entrada *(.cod), onde no início identifica* a versão do ficheiro *(Z2 ou Z5)* de imagem, através do magic number, identifica o número de colunas e linhas e o valor máximo *do índice do dicionário*. De seguida, é reservada memória para a matriz, conforme o número de colunas e linhas.

No final, os valores são colocados na matriz.

Função read_cod_header

Função responsável pela leitura dos dados do cabeçalho do ficheiro de entrada (.cod) (Linhas, colunas, índice ,valor do índice máximo do dicionário) verificando sempre se existem comentários recorrendo à função skip_comments_spaces e, coloca os respetivos valores na estrutura.

Função read_dictionary

Função responsável pela leitura dos dicionários e respetiva alocação dos dados na sua estrutura, recorrendo às funções read_header_dirct para validar o cabeçalho, de seguida carrega os blocos de dados para a estrutura recorrendo a duas funções, allocate_dict_blocks e load_blocks_to struct, fechando o ficheiro e devolvendo a estrutura.

Função read_header_dict

Função responsável pela leitura dos dados do cabeçalho do ficheiro dicionário e respectiva alocação dos dados na respetiva estrutura.

Função load_cod_file_to_struct

Função responsável pelo carregamento do ficheiro *.cod* para respetiva estrutura, usando como função auxiliar skip_comments_spaces para ir validando as linhas do ficheiro.

Função alocate_dict_blocks

Função responsável pela alocação de memória para os blocos do dicionário, recebendo por parâmetros a altura e comprimento dos blocos assim como o número total de blocos, depois através de um ciclo e recorrendo á função malloc() atribui a memória para o tamanho correto de cada bloco.

Função load_blocks_to_struct

Função responsável por carregar os blocos do dicionário para a sua estrutura, de acordo com o número total de locos e altura e largura dos mesmos blocos.

Função dealloc_dict_blocks

Esta função *liberta* espaço na memória para uma matriz composta pelos blocos do dicionário.

Função read_file

Esta função faz a leitura do ficheiro de entrada *(.pgm), onde no início identifica* a versão do ficheiro *(P2 ou P5)* de imagem, através do *magic number*, identifica o número de co*lunas e linhas e o valor máximo de intensidade*. De seguida, *chama as funções correspondentes para reservar* memória para a matriz, conforme o número de colunas e linhas *No final devolve a estrutura.*

Função allocate_matrix

Esta função reserva espaço na memória para uma matriz composta por colunas e linhas, conforme o ficheiro de imagem.

Função dealloc_matrix

Esta função *liberta* espaço na memória para uma matriz composta por colunas e linhas, conforme o ficheiro de imagem.

Função read_header

Função responsável pela leitura dos dados do cabeçalho do ficheiro de entrada (.pgm) (Linhas, colunas, Valor máximo de intensidade) verificando sempre se existem comentários recorrendo à função skip_comments_spaces e, coloca os respetivos valores na estrutura.

Função load_matrix_to_struct_p2

Função responsável por carregar o ficheiro do tipo pgm com o formato P2 (ASCII) para a respetiva matriz, recebendo por referência a matriz, o número de linhas e colunas e o respetivo ficheiro..

Função load_matrix_to_struct_p5

Função responsável por carregar o ficheiro do tipo pgm com o formato P5 (binário) para a repetiva matriz, , recebendo por referência a matriz, o número de linhas e colunas,o respetivo ficheiro e o *max gray value*.

.

Função write_pgm_file

Função responsável por construir o ficheiro *.pgm* (Cabelcalho e matriz), gravando-o na directoria do ficheiro original *.cod*

Funções Principais – ficheiro main.c

Função main

A função *main* consiste na função principal que irá invocar as principais funcionalidades da aplicação, nomeadamente a conversão dos ficheiros de imagem, conforme enunciado a*presentados pelos professores.*

Esta função trata o *parsing* dos parâmetros passados pela linha de comandos.

Este processamento dos parâmetros do utilizador será realizado por intermédio da biblioteca <*cmdline.h*> criada pelo ficheiro de configu*ração gengetopt pagegengetopt*.ggo.

Esta função é também responsável pela contabilização do tempo despendido em cada operação, recorrendo ao uso da biblioteca <time.h> e da função clock().

3 - Conclusão

Este relatório tem como objetivo principal a documentação do código realizado e implementado demonstrando quais das funcionalidades do paCodec estão funcionais e a validação dos argumentos permitidos e/ou obrigatórios solicitados no enunciado.

Os exercícios realizados nas aulas foram uma grande ajuda e os conhecimentos relativos à linguagem tornaram esta etapa aliciante e um grande desafio.

Nem todos os objetivos foram atingidos para as várias funcionalidades, e para as funcionalidades que foram implementas entendemos que a solução atual apresentada, possa não ser a mais correta, uma vez que nesta área, várias opções possam ser tomadas para o mesmo fim, até porque o projeto envolveu um elevado grau de complexidade e a nossa pouca experiência na linguagem C tornou o desenvolvimento desta etapa uma tarefa árdua de concretizar.

As principais dificuldades que encontrámos para a realização desta etapa consistiram na programação do algoritmo de descodificação das imagens e da descodificação recursiva de todos os ficheiros .cod.

A elaboração desta etapa contribuiu para melhorar os nossos conhecimentos como estudantes e programadores, concluindo que a linguagem de programação C é bastante versátil e focalizada na programação procedimental.

4 - Objetivos

Objetivos Atingidos

- Ficheiro Makefile criado e compilado com sucesso
- Ficheiro pagegengetopt.ggo (gengeteopt) para validar parâmetros de entrada criado e compilado com sucesso
- Funcionalidade de descodificar imagens em formato *.cod* do tipo Z2 implementada cm sucesso.
- Funcionalidade do cálculo do PSNR para ficheiros de imagem em formato *.pgm* do tipo P2 implementada com sucesso.
- Criação e gravação do ficheiro descodificado *.pgm* para a diretoria do ficheiro original *.cod* , implementada com sucesso.

Objetivos Não Atingidos

- Na funcionalidade de descodificar (–decode) ficheiros de imagem em formato *.cod*, não conseguimos implementar para os ficheiros do tipo Z5 (binário).
- Na funcionalidade do cálculo do PSNR não conseguimos implementar o cálculo para ficheiros de imagem em formato *.pgm* tipo P5 (binário).
- Na funcionalidade de descodificar (--decode-dir) todos os ficheiros de imagem em formato .cod de uma diretoria dada, não conseguimos implementar na totalidade.

5 - Bibliografia

As referências bibliográficas de seguida, foram de especial auxílio para a criação das caraterísticas do descodificador de imagens necessárias á 1ª etapa.

- Apresentações das aulas teóricas/práticas das aulas de Programação Avançada
- Exercícios/Fichas práticas das aulas de Programação Avançada
- Comando *man* (manual de instruções) usado nas funções passadas na linha de comandos de *Linux*
- *Livro C por "Luís Damas*