Nome Sobrenome do Aluno

Modelo de Relatório AEDs

Brasil

23/05/2017

Nome Sobrenome do Aluno

Modelo de Relatório AEDs

Trabalho apresentado para a Aula Prática N da disciplina de Algoritmos e Estrutura de Dados

Universidade do Norte do Paraná – UNOPAR

Engenharia da Computação

Brasil

23/05/2017

Resumo

Segundo a ABNT (2003, 3.1-3.2), o resumo deve ressaltar o objetivo, o método, os resultados e as conclusões do documento. A ordem e a extensão destes itens dependem do tipo de resumo (informativo ou indicativo) e do tratamento que cada item recebe no documento original. O resumo deve ser precedido da referência do documento, com exceção do resumo inserido no próprio documento. (...) As palavras-chave devem figurar logo abaixo do resumo, antecedidas da expressão Palavras-chave:, separadas entre si por ponto e finalizadas também por ponto.

Palavras-chaves: latex. abntex. editoração de texto.

Sumário

Introdução

Este documento e seu código-fonte são exemplos de referência de uso da classe abntex2 e do pacote abntex2cite. O documento exemplifica a elaboração de relatórios técnicos e/ou científicos produzidos conforme a ABNT NBR 10719:2011 *Informação e documentação - Relatório técnico e/ou científico - Apresentação.*

A expressão "Modelo canônico" é utilizada para indicar que abnTEX2 não é modelo específico de nenhuma universidade ou instituição, mas que implementa tão somente os requisitos das normas da ABNT. Uma lista completa das normas observadas pelo abnTEX2 é apresentada em abnTeX2 e Araujo (2013a).

Sinta-se convidado a participar do projeto abnTEX2! Acesse o site do projeto em http://abntex2.googlecode.com/. Também fique livre para conhecer, estudar, alterar e redistribuir o trabalho do abnTEX2, desde que os arquivos modificados tenham seus nomes alterados e que os créditos sejam dados aos autores originais, nos termos da "The LATEX Project Public License"[1].

Encorajamos que sejam realizadas customizações específicas deste exemplo para universidades e outras instituições — como capas, folhas de rosto, etc. Porém, recomendamos que ao invés de se alterar diretamente os arquivos do abnTEX2, distribua-se arquivos com as respectivas customizações. Isso permite que futuras versões do abnTEX2 não se tornem automaticamente incompatíveis com as customizações promovidas. Consulte abnTeX2 (2013) par mais informações.

Este documento deve ser utilizado como complemento dos manuais do abnTEX2 (ABNTEX2; ARAUJO, 2013a; ABNTEX2; ARAUJO, 2013b; ABNTEX2; ARAUJO, 2013c) e da classe memoir (WILSON; MADSEN, 2010).

Equipe abnTEX2

Lauro César Araujo

[1] http://www.latex-project.org/lppl.txt

1 Desenvolvimento

1.1 Código em Python

Exibindo código fonte na linguagem python usando o pacote Minted (fonte: https://pt.sharelatex.com/learn/Code_Highlighting_with_minted, acesso em 22/05/2017).

```
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None #to become the incidence matrix
    VT = np.zeros((n*m,1), int)  #dummy variable

    #compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m-1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

                if M is None:
                    M = np.copy(VT)
                else:
                    M = np.concatenate((M, VT), 1)

                VT = np.zeros((n*m,1), int)

    return M
```

1.2 Código em C

A seguir apresentamos o Algoritmo de Dijkstra que resolve o problema do caminho mínimo com um vértice-fonte em grafos cujas arestas tenham peso maior ou igual a zero (fonte: https://www.vivaolinux.com.br/script/Algoritmo-de-Dijkstra, acesso em 22/05/2017).

```
1  /*
2  * Este programa implementa o algoritmo de Dijasktra para o problema do
3  * caminho de custo minimo em grafos dirigidos com custos positivos nas
4  * arestas.
5  *
6  * @autor : vanderson lucio
7  * @e-mail: vanderson.gold@gmail.com
8  *
9  */
10
11 #include <stdio.h>
12 #include <stdlib.h>
13 #include <math.h>
14
15 #define FLSH gets(l)
16
17 int destino, origem, vertices = 0;
18 int custo, *custos = NULL;
19
20 void dijkstra(int vertices,int origem,int destino,int *custos)
21 {
22    int i,v, cont = 0;
23    int *ant, *tmp;
24    int *z;     /* vertices para os quais se conhece o caminho minimo */
25    double min;
26    double dist[vertices]; /* vetor com os custos dos caminhos */
27
28
29    /* aloca as linhas da matriz */
30    ant = calloc (vertices, sizeof(int *));
31    tmp = calloc (vertices, sizeof(int *));
32    if (ant == NULL) {
33       printf ("** Erro: Memoria Insuficiente **");
```

```
34          exit(-1);
35       }
36
37       z = calloc (vertices, sizeof(int *));
38       if (z == NULL) {
39          printf ("** Erro: Memoria Insuficiente **");
40          exit(-1);
41       }
42
43       for (i = 0; i < vertices; i++) {
44          if (custos[(origem - 1) * vertices + i] !=- 1) {
45             ant[i] = origem - 1;
46             dist[i] = custos[(origem-1)*vertices+i];
47          }
48          else {
49             ant[i]= -1;
50             dist[i] = HUGE_VAL;
51          }
52          z[i]=0;
53       }
54       z[origem-1] = 1;
55       dist[origem-1] = 0;
56
57       /* Laco principal */
58       do {
59
60          /* Encontrando o vertice que deve entrar em z */
61          min = HUGE_VAL;
62          for (i=0;i<vertices;i++)
63             if (!z[i])
64                if (dist[i]>=0 && dist[i]<min) {
65                   min=dist[i];v=i;
66                }
67
68          /* Calculando as distancias dos novos vizinhos de z */
69          if (min != HUGE_VAL && v != destino - 1) {
70             z[v] = 1;
71             for (i = 0; i < vertices; i++)
72                if (!z[i]) {
```

```
73                 if (custos[v*vertices+i] != -1 && dist[v] +
                   ↪  custos[v*vertices+i] < dist[i]) {
74                       dist[i] = dist[v] + custos[v*vertices+i];
75                    ant[i] =v;
76                    }
77                 }
78        }
79     } while (v != destino - 1 && min != HUGE_VAL);
80
81     /* Mostra o Resultado da busca */
82     printf("\tDe %d para %d: \t", origem, destino);
83     if (min == HUGE_VAL) {
84        printf("Nao Existe\n");
85        printf("\tCusto: \t- \n");
86     }
87     else {
88        i = destino;
89        i = ant[i-1];
90        while (i != -1) {
91        //   printf("<-%d",i+1);
92           tmp[cont] = i+1;
93           cont++;
94           i = ant[i];
95        }
96
97        for (i = cont; i > 0 ; i--) {
98           printf("%d -> ", tmp[i-1]);
99        }
100       printf("%d", destino);
101
102       printf("\n\tCusto:  %d\n",(int) dist[destino-1]);
103    }
104 }
105
106 void limpar(void)
107 {
108      printf("{FONTE}33[2J"); /* limpa a tela */
109      printf("{FONTE}33[1H"); /* poe o curso no topo */
110 }
```

```
111
112 void cabecalho(void)
113 {
114     limpar();
115    printf("Implementacao do Algoritmo de Dijasktra\n");
116    printf("Comandos:\n");
117    printf("\t d - Adicionar um Grafo\n"
118           "\t r - Procura Os Menores Caminhos no Grafo\n"
119           "\t CTRL+c - Sair do programa\n");
120    printf(">>> ");
121 }
122
123 void add(void)
124 {
125    int i, j;
126
127    do {
128       printf("\nInforme o numero de vertices (no minimo 2 ): ");
129       scanf("%d",&vertices);
130    } while (vertices < 2 );
131
132    if (!custos)
133       free(custos);
134    custos = (int *) malloc(sizeof(int)*vertices*vertices);
135    for (i = 0; i <= vertices * vertices; i++)
136       custos[i] = -1;
137
138    printf("Entre com as Arestas:\n");
139    do {
140       do {
141          printf("Origem da aresta (entre 1 e %d ou '0' para sair): ",
           ↪  vertices);
142          scanf("%d",&origem);
143       } while (origem < 0 || origem > vertices);
144
145       if (origem) {
146          do {
147             printf("Destino da aresta (entre 1 e %d, menos %d): ",
              ↪  vertices, origem);
```

```
148             scanf("%d", &destino);
149          } while (destino < 1 || destino > vertices || destino ==
             ↪  origem);
150
151          do {
152             printf("Custo (positivo) da aresta do vertice %d para o
                ↪  vertice %d: ",
153                   origem, destino);
154             scanf("%d",&custo);
155          } while (custo < 0);
156
157          custos[(origem-1) * vertices + destino - 1] = custo;
158       }
159
160    } while (origem);
161 }
162
163 void procurar(void)
164 {
165    int i, j;
166
167    /* Azul */
168    printf("{FONTE}33[36;1m");
169    printf("Lista dos Menores Caminhos no Grafo Dado: \n");
170
171    for (i = 1; i <= vertices; i++) {
172       for (j = 1; j <= vertices; j++)
173          dijkstra(vertices, i,j, custos);
174       printf("\n");
175    }
176
177    printf("<Pressione ENTER para retornar ao menu principal>\n");
178    /* Volta cor nornal */
179    printf("{FONTE}33[m");
180 }
181
182 int main(int argc, char **argv) {
183    int i, j;
184    char opcao[3], l[50];
```

```
185
186     do {
187
188         cabecalho();
189         scanf("%s", &opcao);
190
191         if ((strcmp(opcao, "d")) == 0) {
192             add();
193         }
194         FLSH;
195
196         if ((strcmp(opcao, "r") == 0) && (vertices > 0) ) {
197             procurar();
198             FLSH;
199         }
200
201     } while (opcao != "x");
202
203     printf("\nAte a proxima...\n\n");
204
205     return 0;
206 }
```

Conclusão

Sed consequat tellus et tortor. Ut tempor laoreet quam. Nullam id wisi a libero tristique semper. Nullam nisl massa, rutrum ut, egestas semper, mollis id, leo. Nulla ac massa eu risus blandit mattis. Mauris ut nunc. In hac habitasse platea dictumst. Aliquam eget tortor. Quisque dapibus pede in erat. Nunc enim. In dui nulla, commodo at, consectetuer nec, malesuada nec, elit. Aliquam ornare tellus eu urna. Sed nec metus. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Phasellus id magna. Duis malesuada interdum arcu. Integer metus. Morbi pulvinar pellentesque mi. Suspendisse sed est eu magna molestie egestas. Quisque mi lorem, pulvinar eget, egestas quis, luctus at, ante. Proin auctor vehicula purus. Fusce ac nisl aliquam ante hendrerit pellentesque. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Morbi wisi. Etiam arcu mauris, facilisis sed, eleifend non, nonummy ut, pede. Cras ut lacus tempor metus mollis placerat. Vivamus eu tortor vel metus interdum malesuada.

Sed eleifend, eros sit amet faucibus elementum, urna sapien consectetuer mauris, quis egestas leo justo non risus. Morbi non felis ac libero vulputate fringilla. Mauris libero eros, lacinia non, sodales quis, dapibus porttitor, pede. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Morbi dapibus mauris condimentum nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Etiam sit amet erat. Nulla varius. Etiam tincidunt dui vitae turpis. Donec leo. Morbi vulputate convallis est. Integer aliquet. Pellentesque aliquet sodales urna.

Referências

ABNTEX2. *Como customizar o abnTeX2*. 2013. Wiki do abnTeX2. Disponível em: <https://code.google.com/p/abntex2/wiki/ComoCustomizar>. Acesso em: 23.3.2013. Citado na página 5.

ABNTEX2; ARAUJO, L. C. *A classe abntex2: Modelo canônico de trabalhos acadêmicos brasileiros compatível com as normas ABNT NBR 14724:2011, ABNT NBR 6024:2012 e outras*. [S.l.], 2013. Disponível em: <http://abntex2.googlecode.com/>. Citado na página 5.

ABNTEX2; ARAUJO, L. C. *O pacote abntex2cite: Estilos bibliográficos compatíveis com a ABNT NBR 6023*. [S.l.], 2013. Disponível em: <http://abntex2.googlecode.com/>. Citado na página 5.

ABNTEX2; ARAUJO, L. C. *O pacote abntex2cite: tópicos específicos da ABNT NBR 10520:2002 e o estilo bibliográfico alfabético (sistema autor-data*. [S.l.], 2013. Disponível em: <http://abntex2.googlecode.com/>. Citado na página 5.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. *NBR 6028*: Resumo - apresentação. Rio de Janeiro, 2003. 2 p. Citado na página 2.

WILSON, P.; MADSEN, L. *The Memoir Class for Configurable Typesetting - User Guide*. Normandy Park, WA, 2010. Disponível em: <http://mirrors.ctan.org/macros-/latex/contrib/memoir/memman.pdf>. Acesso em: 19.12.2012. Citado na página 5.