

```
1: program calculo_de_carga;
2: {Este programa realiza um cálculo de carga simplificado para um alto forno, feito com base nos exercícios feitos
3: em sala de aula e, também, com base nos conhecimentos do grupo, tanto na montagem da linha de pensamento com na
4: montagem do programa usando a linguagem Pascal.
5: O programa considera o carregamento de até 3 tipos de minérios (estes podem ser os minérios, pelotas e sinteres).
6: A partir de uma proporção em peso fixa, a carga de minérios é calculada. Calcula-se também o teor médio de ferro
7: contido nela.
8: A carga de coque é feita de forma similar: estipula-se a proporção dos diferentes tipos de coque( no máximo 3)
9: e calcula-se a massa de cada um deles para atender ao processo.
10: Pode-se calcular a quantidade de PCI máxima a se injetar, e a partir dela, estipular um valor a ser usado.
11: O sopro pode ser calculado, considerando-se a necessidade de oxigênio e o seu teor no sopro.
12: Os cálculos são feitos com base nas considerações de que todo o ferro carregado é reduzido
13: e se incorpora ao gusa líquido e a reação de redução da wustita a ferro se processam integralmente a 900°C, de forma
14: que a reação de redução seja:
15:  $\text{FeO} + 3,12\text{CO} = \text{Fe} + 2,12\text{CO} + \text{CO}_2$ .
16: Após o uso do programa, pode-se armazenar os dados em um arquivo texto(Cálculo.doc);}
17:
18: uses crt;{Definição da biblioteca a ser utilizada}
19:
20: {Definição das variáveis utilizadas}
21: var   xt,xk,g,h,MT,po2,NCarr,feg,PCcarr,CRmax,CRmin,Pcoq,PCqu,PCI,
22:       TeorCM,CF,PCiesc,PCIratt,PCIrattmx,FR,Vo2,Vsopro,Vazao,TeorMM,MTC,CRate:real;
23:       t,cont,y,z:integer;
24:       s,resp,c,i,opcao:char;
25:       Texto:text;
26:       PropM: array[1..3] of real;
27:       TeorM: array[1..3] of real;
28:       PropC: array[1..3] of real;
29:       TeorC: array[1..3] of real;
30:       MasM: array[1..3] of real;
31:       MasC: array[1..3] of real;
32:
33: begin{1}
34:   clrscr;
35:   textmode( 1 ); {Modo de apresentação}
36:   TextColor( 15 ); {Cor do texto}
37:   gotoxy(13,10);
38:   writeln('CALCULO DE CARGA');
39:   gotoxy(1,24);
40:   write('Pressione qualquer tecla para continuar.');
```

41: readkey;

42: cont:=1;

43: while cont <=40 do {Loop para funções gráficas de apresentação do programa}

44: begin{2}

```
45: write(#220);
46: cont:=cont+1;
47: delay(30);
48: end;{2}
49: Textmode(3);
50: TextColor(15);
51: resp:= 's';
52: while resp ='s' do {Loop principal do programa. Através dele pode-se executar o programa diversas vezes}
53: begin{3}
54:   {Inicialização das variáveis, evitando-se erros nas equações}
55:   xt:=0; {Massa de gusa (em toneladas)}
56:   xk:=0; {Massa de gusa (em kilogramas)}
57:   g:=0; {Teor de ferro no gusa(%)}
58:   h:=0; {Teor de carbono no gusa(%)}
59:   MT:=0; {Massa total de minérios carregados}
60:   po2:=0; {Percentual de oxigênio no ar soprado}
61:   NCcarr:=0; {Número de kmols de carbono necessário para ser carregado}
62:   PCcarr:=0; {Massa de carbono a ser carregado}
63:   CRmax:=0; {Coke rate máximo(em kg/ton de gusa)}
64:   CRmin:=0; {Coke rate mínimo(em kg/ton de gusa)}
65:   Pcoq:=0; {Massa de coque (sem o uso de PCI)-para o cálculo do Coke rate máximo}
66:   PCqu:=0; {Massa de carbono queimado}
67:   PCI:=0; {Massa máxima de PCI que pode ser adicionada}
68:   TeorCM:=0; {Teor médio de carbono fixo na carga de coques}
69:   CF:=0; {Teor de carbono no PCI}
70:   PCiesc:=0; {Massa de PCI, inferior à máxima massa permitida}
71:   PCIrat:=0; {PCI rate (em kg/ton de gusa)}
72:   PCIratmx:=0; {PCI rate máximo (em kg/ton de gusa)}
73:   FR:=0; {Fuel rate (em kg/ton de gusa)}
74:   Vo2:=0; {Volume de oxigênio consumido(em Nm3/dia)}
75:   Vsopro:=0; {Volume de sopro {em Nm3/dia}}
76:   Vazao:=0; {Vazão do sopro(em Nm3/min)}
77:   TeorMM:=0; {Teor médio de ferro na carga dos minérios}
78:   MTC:=0; {Massa total de coque carregado}
79:   CRate:=0; {Coque rate (em kg/ton de gusa)}
80:   cont:=1;
81:   while cont <=3 do { Loop para inicialização dos vetores criados}
82:   begin{4}
83:     PropM[cont]:=0;{Proporção do minério (1,2 ou 3) na carga de minérios}
84:     TeorM[cont]:=0; {Teor de ferro no minério (1,2 ou 3)}
85:     MasM[cont]:=0; {Massa do minério (1,2 ou 3) carregado}
86:     PropC[cont]:=0; {Proporção do coque (1,2 ou 3)}
87:     TeorC[cont]:=0; {Teor de carbono fixo no coque (1,2 ou 3)}
88:     MasC[cont]:=0; {Massa do coque (1,2 ou 3) carregado}
```

```
89:     cont:=cont+1;
90:     end{4};
91:     clrscr;
92:     writeln;
93:     textcolor(12);
94:     writeln('                QUANTIDADE E COMPOSICAO DO GUSA'); {Início da entrada de dados}
95:     textcolor(15);
96:     writeln;
97:     writeln;
98:     Write('Informe a quantidade de gusa que deve ser produzido (em toneladas):');
99:     readln(xt); {Leitura da massa de gusa em toneladas}
100:    writeln;
101:    repeat {Loop de detecção de erros na execução}
102:        writeln('Informe o teor de ferro e carbono no gusa que ser',#160,' produzido. ');
103:        write('Teor de ferro:');
104:        readln(g); {Leitura do teor de ferro no gusa}
105:        write('Teor de carbono:');
106:        readln(h);
107:        writeln;
108:        If g+h>100 then
109:            begin{5}
110:                textcolor(11);
111:                writeln('Dados incorretos!'); {Aviso de erro na digitação dos teores de ferro e carbono}
112:                textcolor(15);
113:            end{5};
114:        until g+h<=100;
115:        clrscr;
116:        textcolor(12);
117:        writeln('                TIPOS DE MINERIO');
118:        textcolor(15);
119:        writeln;
120:        writeln;
121:        repeat
122:            Writeln('Quantos tipos de min,rios serao utilizados(m',#160,'ximo de 3)?');
123:            readln(t); {Leitura do número de minérios (ou similares) a serem utilizados}
124:            writeln;
125:            if t > 3 then
126:                begin{6}
127:                    textcolor(11);
128:                    writeln('Dado incorreto!'); {Aviso de erro na digitação do número de minérios}
129:                    textcolor(15);
130:                end{6};
131:            until t<=3;
132:        repeat
```

```
133:      writeln;
134:      textcolor(12);
135:      writeln;
136:      writeln('                PROPORCAO DE MINERIO E TEOR DE FERRO');
137:      textcolor(15);
138:      writeln;
139:      writeln;
140:      repeat {Loop de leitura da proporção do minério na carga e do teor de ferro nele}
141:      if t=1 then
142:      begin{7}
143:          writeln('Informe o teor de ferro no minerio. ');
144:          readln(TeorM[1]); {Leitura do teor de ferro, no caso de um só minério}
145:          PropM[1]:=100;
146:      end{7}
147:      else
148:      begin{8}
149:          cont:=1;
150:          while cont<=t do {Leitura da proporção e do teor de ferro para cada minério, para o caso de mais de um minério}
151:          begin{9}
152:              writeln('Informe a proporcao do minerio ',cont,' e o teor de ferro nele. ');
153:              write('Proporcao: ');
154:              readln(PropM[cont]); {Leitura da proporção para cada minério}
155:              write('Teor de ferro: ');
156:              readln(TeorM[cont]); {Leitura do teor de ferro para cada minério}
157:              cont:=cont+1;
158:              writeln;
159:          end{9};
160:      end{8};
161:      if PropM[1]+PropM[2]+PropM[3] <> 100 then
162:      begin{10}
163:          textcolor(11);
164:          writeln('Dados incorretos!'); {Aviso de erro na digitação das proporções}
165:          textcolor(15);
166:      end{10};
167:      until PropM[1]+PropM[2]+PropM[3]=100;
168:      TeorMM:=(TeorM[1]*PropM[1]+TeorM[2]*PropM[2]+TeorM[3]*PropM[3])/100;
169:      clrscr;
170:      textcolor(12);
171:      writeln('                TIPOS DE COQUE');
172:      textcolor(15);
173:      writeln;
174:      writeln;
175:      repeat
176:          writeln('Quantos tipos de coque serao utilizados(m',#160,'ximo de 3)?');
```

```
177:     readln(y); {Leitura do número de coques a serem utilizados}
178:     writeln;
179:     if y > 3 then
180:     begin{11}
181:         textcolor(11);
182:         writeln('Dado incorreto!'); {Aviso de erro na digitação do número de coques}
183:         textcolor(15);
184:     end{11};
185: until y<=3;
186: writeln;
187: writeln;
188: textcolor(12);
189: writeln('                PROPORCAO E TEOR DE CARBONO');
190: textcolor(15);
191: writeln;
192: repeat
193:     if y=1 then
194:     begin{12}
195:         writeln('Informe o teor de carbono no coque. ');
196:         readln(TeorC[1]); {Leitura do teor de carbono fixo, no caso de um só coque}
197:         PropC[1]:=100;
198:     end{12}
199: else
200:     begin{13}
201:         cont:=1;
202:         while cont <=y do {Leitura da proporção e do teor de carbono para cada coque, para o caso de mais de um coque}
203:         begin{14}
204:             writeln('Informe a propor',#135,'ao do coque ',cont,' na mistura e teor de carbono fixo nele. ');
205:             write('Propor',#135,'ao: ');
206:             readln(PropC[cont]); {Leitura da proporção para cada coque}
207:             write('Teor de carbono: ');
208:             readln(TeorC[cont]); {Leitura do teor de carbono para cada coque}
209:             cont:=cont+1;
210:             writeln;
211:         end{14};
212:     end{13};
213: if PropC[1]+PropC[2]+PropC[3] <> 100 then
214:     begin{15}
215:         textcolor(11);
216:         writeln('Dados incorretos!'); {Aviso de erro na digitação das proporções}
217:         textcolor(15);
218:     end{15};
219: until PropC[1]+PropC[2]+PropC[3]=100;
220: clrscr;
```

```
221:   xk:= xt*1000;{Conversão de unidades para a massa de gusa: de toneladas para kilogramas}
222:   feg:= (xk*g)/(100);{Cálculo da massa de ferro no gusa (feg),em kilogramas} }
223:   MT:= (feg*10000)/(TeorM[1]*PropM[1]+TeorM[2]*PropM[2]+TeorM[3]*PropM[3]); {Cálculo da massa total de minérios
      carregados (em kilogramas)}
224:   cont:=1;
225:   while cont<= 3 do
226:     begin{16}
227:       MasM[cont]:= (PropM[cont]*MT/100);{Cálculo da massa individual de cada minério ( em kilogramas)}
228:       MasM[cont]:=MasM[cont]/1000;{Conversão da massa individual de cada minério de kilogramas para toneladas}
229:       cont:= cont+1;
230:     end{16};
231:   cont:=1;
232:   NCcarr:=(2.12*feg)/(56)+(xk*h)/(1200); {Cálculo do número de kmols de carbono a serem carregados}
233:   PCcarr:=NCcarr*12; {Cálculo da massa de carbono a ser carregada (em kilogramas)}
234:   TeorCM:=(TeorC[1]*PropC[1]+TeorC[2]*PropC[2]+TeorC[3]*PropC[3])/100; {Cálculo do teor médio de carbono fixo na
      mistura de coques}
235:   Pcoq:= PCcarr/(TeorCM/100);{peso de coque sem uso de PCI }
236:   cont:=1;
237:   CRmax:=Pcoq/xt; {Cálculo do coke rate máximo (em kilogramas/ ton de gusa)}
238:   PCqu:=(0.02*feg)*(12); {Cálculo da massa de carbono queimado (em kilogramas)}
239:   writeln;
240:   textcolor(12);
241:   Writeln('                                QUANTIDADE DE PCI');
242:   textcolor(15);
243:   writeln;
244:   writeln;
245:   writeln('Informe o teor de carbono no PCI:');
246:   readln(CF); {Leitura do teor de carbono fixo no PCI}
247:   PCI:=(PCqu)/(CF/100); {Cálculo da massa máxima de PCI permitida (em kilogramas)}
248:   PCIratmx:= PCI/xt; {Cálculo do PCI rate máximo (em kilogramas/ton de gusa)}
249:   PCI:=PCI/1000; {Conversão da massa máxima de PCI de kilogramas para toneladas}
250:   writeln;
251:   repeat
252:     writeln('Voc^ poder  usar no m ximo ',PCI:10:3,' toneladas de PCI. ');
253:     write('Informe um valor em toneladas:');
254:     readln(PCIesc); {Leitura da massa de PCI adicionada}
255:     writeln;
256:     if PCIesc > PCI then
257:       begin{17}
258:         textcolor(11);
259:         writeln('Dado incorreto!'); {Aviso de digitação de valor incorreto}
260:         textcolor(15);
261:       end{17};
262:   until PCIesc <= PCI;
```

```
263:   PCIesc:=PCIesc*1000; {Conversão da massa de PCI adicionada de toneladas para kilogramas}
264:   MTC:=(24*feg-PCIesc*CF+feg*150/7+xk*h)/(TeorCM); {Massa total da mistura de coque a ser carregada (em kilogramas)}
265:   CRate:=MTC/xt; {Coke rate (em kilogramas/ton de gusa)}
266:   CRmin:=(((feg/56+(xk*h)/1200)*12)*100)/(xt*TeorCM); {Cálculo do coke rate mínimo (em kilogrmas/ton de gusa)}
267:   cont:=1;
268:   while cont<= 3 do
269:     begin{18}
270:       MasC[cont]:= (PropC[cont]*(MTC/1000))/(100); {Cálculo da massa ndividual de cada coque carregado (em toneladas)}
271:       cont:= cont+1;
272:     end{18};
273:   PCIrat:=PCIesc/xt; {Cálculo do PCI rate (em kilogramas/ton de gusa)}
274:   FR:= PCIrat + CRate; {Cálculo do Fuel rate (em kilogramas/ton de gusa)}
275:   PCIesc:=PCIesc/1000; {Conversão da massa de PCI adicionada de kilogramas para toneladas}
276:   clrscr;
277:   textcolor(12);
278:   writeln('
                                     TEOR DE OXIGENIO NO SOPRO');
279:   textcolor(15);
280:   writeln;
281:   writeln;
282:   writeln('Informe o teor de oxigenio no sopro. ');
283:   write('Teor de oxigenio: ');
284:   readln(po2); {Leitura do teor de oxigênio no sopro}
285:   Vo2:= 0.224*feg; {Cálculo do volume}????????????????????????????????????????????????????????????
   ?????????????????????????????????????????
286:   Vsopro:= Vo2/(po2/100); {Cálculo do volume de ar a ser soprado (em Nm3)}
287:   Vazao:= Vsopro/1440; {Cálculo da vazão de sopro (em Nm3/min)}
288:   clrscr;
289:   {Apresentação dos dados de entrada e dos resultados}
290:   {Apresentação dos dados de entrada}
291:   cont:=1;
292:   while cont<=80 do {Loop para funções gráficas de apresentação dos dados de entrada}
293:     begin{19}
294:       write(#205);
295:       cont:=cont+1;
296:     end{19};
297:   cont:=1;
298:   while cont<=80 do {Loop para funções gráficas de apresentação dos dados de entrada}
299:     begin{20}
300:       gotoxy(cont,3);
301:       write(#205);
302:       cont:=cont+1;
303:     end{20};
304:   cont:=1;
305:   while cont<=79 do {Loop para funções gráficas de apresentação dos dados de entrada}
```

```
306: begin{21}
307:   gotoxy(cont,25);
308:   write(#205);
309:   cont:=cont+1;
310: end{21};
311: gotoxy(35,2);
312: textcolor(12);
313: writeln('DADOS DE ENTRADA');
314: gotoxy(1,4);
315: write('-----GUSA:-----');
316: textcolor(15);
317: writeln('Gusa:',xt:7:2,' ton (' ,g:5:2,' % de Fe,' ,h:5:2,' % de C.).'); {Apresentação da massa de gusa produzida
318: e dos teores presentes de ferro e carbono}
319: textcolor(12);
320: writeln;
321: writeln('-----MINERIOS:-----');
322: textcolor(15);
323: writeln('Minerio          Proporcao(%)          Teor(%)');
324: cont:=1;
325: while cont <=t do
326:   begin{22}
327:     writeln('Minerio ',cont,'          ',PropM[cont]:7:2,'          ',TeorM[cont]:7:2); {Apresentação da proporçã
328: o de cada minério na carga e do respectivo teor de ferro}
329:   end{22};
330:   writeln('          TEOR MEDIO DE FERRO: ',TeorMM:3:2,'%'); {Apresentação do teor médio de ferro na
331:   carga}
332:   textcolor(12);
333:   writeln('-----COQUE:-----');
334:   textcolor(15);
335:   writeln('Coque          Proporcao(%)          Teor(%)');
336:   cont:=1;
337:   while cont <=y do
338:     begin{23}
339:       writeln('Coque ',cont,'          ',PropC[cont]:7:2,'          ',TeorC[cont]:7:2); {Apresentação da proporção
340: de cada coque na carga e do respectivo teor de carbono fixo}
341:     end{23};
342:     writeln('          TEOR MEDIO DE CARBONO: ',TeorCM:3:2,'%'); {Apresentação do teor médio de carbono fixo
343:     na carga}
344:     writeln;
345:     writeln('TEOR DE CARBONO NO PCI: ',CF:3:2,'%'); {Apresentação do teor de carbono fixo do PCI}
346:     writeln('TEOR DE OXIGENIO NO SOPRO: ',po2:3:2,'%'); {Apresentação do teor de oxigênio no sopro}
```



```
346:      writeln('QUANTIDADE DE PCI ADICIONADA: ',PCIesc:7:2,' ton. '); {Apresentação da quantidade de PCI adicionada}
347:      gotoxy(80,25);
348:      readkey;
349:      clrscr;
350:      {Apresentação dos resultados}
351:      cont:=1;
352:      while cont<=80 do {Loop para funções gráficas de apresentação dos resultados}
353:      begin{24}
354:          write(#205);
355:          cont:=cont+1;
356:      end{24};
357:      cont:=1;
358:      while cont<=80 do {Loop para funções gráficas de apresentação dos resultados}
359:      begin{25}
360:          gotoxy(cont,3);
361:          write(#205);
362:          cont:=cont+1;
363:      end{25};
364:      cont:=1;
365:      while cont<=80 do {Loop para funções gráficas de apresentação dos resultados}
366:      begin{26}
367:          gotoxy(cont,24);
368:          write(#205);
369:          cont:=cont+1;
370:      end{26};
371:      gotoxy(35,2);
372:      textcolor(12);
373:      write('RESULTADOS');
374:      textcolor(15);
375:      gotoxy(1,4);
376:      cont:=1;
377:      while cont<= t do
378:      begin{27}
379:          writeln('Massa de minerio ',cont,': ',MasM[cont]:7:2,' ton. '); {Apresentação da massa de cada minério carregado}
380:          cont:=cont+1;
381:      end{27};
382:      MT:=MT/1000; {Conversão da massa total de minério carregado de kilogramas para toneladas}
383:      textcolor(12);
384:      writeln('TOTAL CARREGADO: ',MT:7:2,' toneladas por dia. '); {Massa total de minério carregado diariamente}
385:      textcolor(15);
386:      cont:=1;
387:      while cont<= y do
388:      begin{28}
```

```
389:         writeln('Massa do coque ',cont,': ',MasC[cont]:7:2,' ton. '); {Apresentação da massa de cada coque carregado}
390:         cont:=cont+1;
391:     end{ 28};
392: {     end{ };}????????????????????????????????????????????????????????????????????????????????????????????
    ??????????
393:     MTC:=MTC/1000;
394:     textcolor(12);
395:     writeln('TOTAL CARREGADO: ',MTC:7:2,' toneladas por dia. '); {Massa total de coque carregado diariamente}
396:     textcolor(15);
397:     textcolor(12);
398:     textcolor(12);
399:     writeln('Coke Rate: ', CRate:7:2,' kg/ton de gusa'); {Apresentação do Coke rate obtido}
400:     textcolor(15);
401:     writeln('Coke Rate maximo: ',CRmax:7:2,' kg/ton de gusa'); {Apresentação do Coke rate máximo (em uma operação ALL
        COKE)}
402:     writeln('Coke Rate minimo: ',CRmin:7:2,' kg/ton de gusa'); {Apresentação do Coke rate teórico mínimo
        (através da utilização da quantidade máxima de PCI)}
403:     writeln;
404:     textcolor(12);
405:     writeln('Massa do PCI: ',PCIesc:5:2,' ton'); {Apresentação da massa de PCI adicionada}
406:     writeln('PCI Rate: ',PCIrat :7:2,' kg/ton de gusa'); {Apresentação do PCI rate obtido}
407:     textcolor(15);
408:     writeln('PCI Rate maximo: ',PCIratmx:5:2,' Kg/ton de gusa'); {Apresentação do PCI rate máximo permitido}
409:     textcolor(12);
410:     writeln;
411:     writeln('Fuel Rate: ',FR:7:2,' kg/ton de gusa'); {Apresentação do Fuel rate obtido}
412:     writeln;
413:     textcolor(15);
414:     writeln('Volume de oxigenio consumido: ',Vo2:5:2,' Nm3/dia. '); {Apresentação do volume de oxigênio consumido}
415:     textcolor(12);
416:     writeln('Volume de sopro: ',Vazao:5:2,' Nm3/min'); {Apresentação do vazão de sopro}
417:     textcolor(15);
418:     gotoxy(1,25);
419:     write('Pressione [Enter] para ir a tela de gravacao de resultados. '); {Finalização da apresentação dos dados de
        entrada e dos resultados}
420:     readln;
421:     clrscr;
422:
423:
424:     {Procedimentos para armazenamento dos dados em um arquivo texto}
425:     Writeln('                GRAVACAO DOS RESULTADOS');
426:     writeln;
427:     writeln;
428:     writeln('Deseja criar um arquivo texto com os dados obtidos?');
429:     write('Opcoes:(s)sim (n)nao?');
```

```
430: readln(resp);
431: writeln;
432: writeln;
433: if resp = 's' then
434:   begin{29}
435:     assign(Texto, 'Calculo.doc');
436:     opcao:='i';
437:     writeln('Sera criado um arquivo chamado: Calculo.doc.');
```

438: writeln;

439: textcolor(12);

440: writeln('ATENCAO! Se esse arquivo j havia sido criado anteriormente e voce escolher a');

441: Writeln('opcao (c)criar,na proxima pergunta, ele ser sobrescrito e TODOS os dados serao');

442: writeln('perdidos.');

443: writeln;

444: textcolor(15);

445: writeln('Deseja criar um arquivo ou continuar inserindo dados no arquivo atual?');

446: writeln('Opcoes:(c)criar (i)inserir:');

447: readln(opcao);

448: if opcao = 'c' then

449: begin{30}

450: rewrite(Texto);

451: end{30}

452: else

453: begin{31}

454: append(Texto);

455: end{31};

456: {Formatação da apresentação dos dados de entrada e dos resultados no arquivo texto}

457: writeln(Texto, ' DADOS DE ENTRADA');

458: writeln(Texto);

459: write(Texto, '-----GUSA:-----');

460: writeln(Texto, ' Gusa:',xt:7:2,' ton (' ,g:5:2,' % de Fe,' ,h:5:2,' % de C.).');

461: writeln(Texto);

462: writeln(Texto, '-----MINERIOS:-----');

463: writeln(Texto);

464: writeln(Texto, ' Minerio Proporcão(%) Teor(%)');

465: cont:=1;

466: while cont <=t do

467: begin{32}

468: writeln(Texto, ' Minerio ',cont, ' ',PropM[cont]:7:2, ' ',TeorM[cont]:7:2);

469: cont:=cont+1;

470: end{32};

471: writeln(Texto, ' TEOR MEDIO DE FERRO: ',TeorMM:3:2,'%');

472: writeln(Texto);

473: writeln(Texto, '-----COQUE:-----');

```

474:      writeln(Texto);
475:      writeln(Texto,'                Coque                Proporcao(%)                Teor(%)');
476:      cont:=1;
477:      while cont <=y do
478:      begin{33}
479:          writeln(Texto,'                Coque ',cont,', '                ',PropC[cont]:7:2,', '                ',TeorC[cont]:7:2);
480:          cont:=cont+1;
481:      end{33};
482:      writeln(Texto,'                                TEOR MEDIO DE CARBONO: ',TeorCM:3:2,'%');
483:      writeln(Texto);
484:      writeln(Texto,'                                TEOR DE CARBONO NO PCI: ',CF:3:2,'%');
485:      writeln(Texto,'                                TEOR DE OXIGENIO NO SOPRO: ',po2:3:2,'%');
486:      writeln(Texto,'                                Quantidade de PCI adicionada: ',PCIesc:7:2,' ton.');
```

```
518:      writeln(Texto, '          Coke Rate maximo: ', CRmax:7:2, ' kg/ton de gusa');
519:      writeln(Texto, '          Coke Rate minimo: ', CRmin:7:2, ' kg/ton de gusa');
520:      writeln(Texto);
521:
522:      writeln(Texto, '          PCI Rate: ', PCIrat :7:2, ' kg/ton de gusa');
523:      writeln(Texto, '          PCI Rate maximo: ', PCIratmx:5:2, ' Kg/ton de gusa');
524:      writeln(Texto);
525:      writeln(Texto, '          Fuel Rate: ', FR:7:2, ' kg/ton de gusa');
526:      writeln(Texto);
527:      writeln(Texto, '-----DADOS DE SOPRO-----');
528:      writeln(Texto, '          Volume de oxigenio consumido: ', Vo2:5:2, ' Nm3/dia. ');
529:      writeln(Texto, '          Volume de sopro: ', Vazao:5:2, ' Nm3/min ');
530:      writeln(Texto);
531:      writeln(Texto);
532:      writeln(Texto);
533:      cont:=2*t+2*y;
534:      while cont<12 do
535:      begin{36}
536:          writeln(Texto);
537:          cont:=cont+1;
538:      end{36};
539:
540:      if opcao='c'then
541:      begin{37}
542:          writeln('O arquivo foi criado com sucesso!');
543:      end{37}
544:      else
545:      begin{38}
546:          writeln('A insercao de dados foi feita com sucesso!');
547:      end{38};
548:      close(Texto);
549:      end{29};
550:      writeln;
551:      writeln;
552:      writeln('Deseja executar o programa novamente?');
553:      write('Opcoes: (s)sim (n)nao?');
554:      readln(resp);
555:      end{3};{end do while inicial}
556:      clrscr;
557:      writeln('          Este programa foi elaborado por alunos do quarto periodo, do curso ');
558:      writeln('          de Tecnologia em Metalurgia e Materiais - CEFET/ES, Abril de 2006. ');
559:      gotoxy(1,5);
560:      textcolor(11);
561:      writeln('          ALUNOS: ');
```

```
562:   writeln;
563:   writeln('           Daniel Figueiredo Leao Oliveira');
564:   writeln;
565:   writeln('           Dickson Alves de Souza');
566:   writeln;
567:   writeln('           Eduardo Junca');
568:   writeln;
569:   writeln('           Erasmo Schultz');
570:   writeln;
571:   writeln('           Gustavo Coqui Barbosa');
572:   writeln;
573:   writeln('           Maria Maria Cordeiro Moreira Cunillo Zacchei');
574:   writeln;
575:   writeln('           Victor Bridi Telles');
576:   writeln;
577:   writeln;
578:   writeln('PROFESSOR: Marcelo Lucas Pereira Machado');
579:   gotoxy(1,25);
580:   textcolor(7);
581:   textbackground(4);
582:   write('Pressione qualquer tecla para sair do programa.');
```

```
583:   readkey;
584: end.{1}
```