```
 1: program TrabalhoComputacional2;
 2:
 3: uses crt;
 4:
 5: var  operacao   :        Integer;         //Operações com matrizes
 6:      op1        :        Integer;         //Operandos 1 e 2 para as operações
 7:      op2        :        Integer;
 8:      k          :        Integer;         //i,j e k - contadores dos lacos for
 9:      i          :        Integer;
10:      j          :        Integer;
11:      vr         :        Integer;         //vr - valor constante p/ operação 3
12:      vet1       :        Array [1..2] of Integer;
13:      vet2       :        Array [1..2] of Integer;     // Vet1 a Vet7 - vetores
14:      vet3       :        Array [1..2] of Integer;     // que armazenam as ordens
15:      vet4       :        Array [1..2] of Integer;     // das matrizes
16:      vet5       :        Array [1..2] of Integer;
17:      vet6       :        Array [1..2] of Integer;
18:      vet7       :        Array [1..2] of Integer;
19:      aux1       :        Array [1..2] of Integer;     // Vetores auxiliares - auxiliam nas
    estruturas
20:      aux2       :        Array [1..2] of Integer;     // condicionais, copiando as ordens das
     matrizes.
21:      matA       :        Array [1..3,1..3] of Integer;
22:      matB       :        Array [1..3,1..4] of Integer;    // MatA a MatG - matrizes de preset
23:      matC       :        Array [1..3,1..1] of Integer;
24:      matD       :        Array [1..1,1..3] of Integer;
25:      matE       :        Array [1..3,1..4] of Integer;
26:      matF       :        Array [1..4,1..2] of Integer;
27:      matG       :        Array [1..3,1..3] of Integer;
28:      mop1       :        Array [1..5,1..5] of Integer;    // mop1 e mop2 - Auxiliam na
    realização de
29:      mop2       :        Array [1..5,1..5] of Integer;    // operações, copiando as matrizes
    de preset
30:      matS       :        Array [1..5,1..5] of Integer;
31:      matSub     :        Array [1..5,1..5] of Integer;    // MatS, MatSub, Matm2k e MatP são
    as
32:      matm2k     :        Array [1..5,1..5] of Integer;    // matrizes resultado das operações
33:      matP       :        Array [1..5,1..5] of Integer;
34:      continue   :        Char;                            // Continuar ou encerrar as
    atividades
35: begin
36:   //Declaração das matrizes de preset do exercício.
37:
38:   //Matriz A 3x3.
39:   matA[1,1] := 5;
40:   matA[1,2] := 2;
41:   matA[1,3] := 1;
42:   matA[2,1] := 15;
43:   matA[2,2] := 7;
44:   matA[2,3] := 7;
45:   matA[3,1] := 25;
46:   matA[3,2] := 7;
47:   matA[3,3] := 5;
48:
49:   //Matriz B 3x4.
50:   matB[1,1] := -5;
51:   matB[1,2] := 2;
52:   matB[1,3] := 1;
53:   matB[1,4] := 1;
54:   matB[2,1] := 15;
55:   matB[2,2] := -7;
56:   matB[2,3] := 7;
57:   matB[2,4] := -1;
```

```
 58:   matB[3,1] := 25;
 59:   matB[3,2] := 7;
 60:   matB[3,3] := 5;
 61:   matB[3,4] := 3;
 62:
 63:   //Matriz C 3x1.
 64:   matC[1,1] := -2;
 65:   matC[2,1] := 1;
 66:   matC[3,1] := 4;
 67:
 68:   //Matriz D 1x3.
 69:   matD[1,1] := -2;
 70:   matD[1,2] := 0;
 71:   matD[1,3] := 5;
 72:
 73:   //Matriz E 3x4.
 74:   matE[1,1] := -5;
 75:   matE[1,2] := 2;
 76:   matE[1,3] := 1;
 77:   matE[1,4] := 1;
 78:   matE[2,1] := 15;
 79:   matE[2,2] := -7;
 80:   matE[2,3] := 7;
 81:   matE[2,4] := -1;
 82:   matE[3,1] := 25;
 83:   matE[3,2] := 7;
 84:   matE[3,3] := 5;
 85:   matE[3,4] := 3;
 86:
 87:   //Matriz F 4x2.
 88:   matF[1,1] := -2;
 89:   matF[1,2] := 0;
 90:   matF[2,1] := 1;
 91:   matF[2,2] := -1;
 92:   matF[3,1] := 5;
 93:   matF[3,2] := -4;
 94:   matF[4,1] := 12;
 95:   matF[4,2] := 3;
 96:
 97:   //Matriz G 3x3.
 98:   matG[1,1] := -2;
 99:   matG[1,2] := -0;
100:   matG[1,3] := 2;
101:   matG[2,1] := 1;
102:   matG[2,2] := -1;
103:   matG[2,3] := 1;
104:   matG[3,1] := 5;
105:   matG[3,2] := -4;
106:   matG[3,3] := 0;
107:
108:   writeln('Bem vindo!');
109:   delay(1000);
110:   writeln('Este programa realiza calculos com matrizes predefinidas');
111:   writeln('Digite qualquer tecla para comecar');
112:   readkey;
113:   clrscr;                                //Limpa a tela
114:   continue := 'y';
115:   delay(200);
116:   while continue <> 'n' do
117:   begin
118:       writeln('Escolha a operacao, ou 5 para sair: ');
119:       writeln('1 - Adicao de matrizes');
120:       writeln('2 - Subtracao de matrizes');
```

```
121:        writeln('3 - Multiplicacao por valor constante');
122:        writeln('4 - Multiplicacao de matrizes');
123:        writeln('5 - Sair');
124:        writeln;
125:        write('Digite a opcao: ');
126:        read(operacao);
127:        clrscr;                             //Limpa a tela
128:        writeln('Exibindo as matrizes: ');
129:        writeln('Escolha as opcoes de 1 a 7 para os operandos conforme solicitado');
130:        writeln;
131:        // Matriz A.
132:        write('1 - A 3x3 =(');
133:        for i := 1 to 3 do
134:            for j := 1 to 3 do
135:                write('a',i,j,':',matA[i,j],'  ');
136:        write(')');
137:        vet1[1] := i;                                   // Esses vetores e os semelhantes armazenam
138:        vet1[2] := j;                                   // as ordens das matrizes para utilizar nas
139:        writeln('Ordem =',vet1[1],vet1[2]);             // estruturas condicionais das operações de
140:        writeln;                                        // adição, subtração e multiplicação entre
141:        // Matriz B.                                    // matrizes.
142:        write('2 - B 3x4 =(');
143:        for i := 1 to 3 do
144:            for j := 1 to 4 do
145:                write('b',i,j,':',matB[i,j],'  ');
146:        write(')');
147:        vet2[1] := i;
148:        vet2[2] := j;
149:        writeln('Ordem =',vet2[1],vet2[2]);
150:        writeln;
151:        // Matriz C.
152:        write('3 - C 3x1 =(');
153:        for i := 1 to 3 do
154:            for j := 1 to 1 do
155:                write('c',i,j,':',matC[i,j],'  ');
156:        write(')');
157:        vet3[1] := i;
158:        vet3[2] := j;
159:        writeln('Ordem =',vet3[1],vet3[2]);
160:        writeln;
161:        // Matriz D.
162:        write('4 - D 1x3 =(');
163:        for i := 1 to 1 do
164:            for j := 1 to 3 do
165:                write('d',i,j,':',matD[i,j],'  ');
166:        write(')');
167:        vet4[1] := i;
168:        vet4[2] := j;
169:        writeln('Ordem =',vet4[1],vet4[2]);
170:        writeln;
171:        // Matriz E.
172:        write('5 - E 3x4 =(');
173:        for i := 1 to 3 do
174:            for j := 1 to 4 do
175:                write('e',i,j,':',matE[i,j],'  ');
176:        write(')');
177:        vet5[1] := i;
178:        vet5[2] := j;
179:        writeln('Ordem =',vet5[1],vet5[2]);
180:        writeln;
181:        // Matriz F.
182:        write('6 - F 4x2 =(');
183:        for i := 1 to 4 do
```

```
184:          for j := 1 to 2 do
185:              write('f',i,j,':',matF[i,j],'  ');
186:      write(')');
187:      vet6[1] := i;
188:      vet6[2] := j;
189:      writeln('Ordem =',vet6[1],vet6[2]);
190:      writeln;
191:      // Matriz G.
192:      write('7 - G 3x3 =(');
193:      for i := 1 to 3 do
194:          for j := 1 to 3 do
195:              write('g',i,j,':',matG[i,j],'  ');
196:      write(')');
197:      vet7[1] := i;
198:      vet7[2] := j;
199:      writeln('Ordem =',vet7[1],vet7[2]);
200:      writeln;
201:      writeln('Pressione qualquer tecla para comecar a escolher as matrizes');
202:      readkey;                //checkpoint
203:      case operacao of
204:           1 : begin                                  // Aqui começa a adição de matrizes
205:                 writeln('Adicao de matrizes');
206:                 writeln;
207:                 write('Escolha a primeira matriz: ');          // 1° operando
208:                 read(op1);
209:                 case op1 of
210:                      1 : begin       //Matriz A 3x3
211:                            for i := 1 to 3 do
212:                                for j := 1 to 3 do
213:                                    mop1[i,j] := matA[i,j];
214:                            for i := 1 to 3 do
215:                                for j := 1 to 3 do
216:                                    write('mop1 ',i,j,':',mop1[i,j],'  ');
217:                            writeln('A ',vet1[1],vet1[2]);
218:                            aux1[1] := vet1[1];
219:                            aux1[2] := vet1[2];
220:                            writeln('Copia 1: ',aux1[1],aux1[2]);
221:                          end;
222:                      2 : begin       //Matriz B 3x4
223:                            for i := 1 to 3 do
224:                                for j := 1 to 4 do
225:                                    mop1[i,j] := matB[i,j];
226:                            for i := 1 to 3 do
227:                                for j := 1 to 4 do
228:                                    write('mop1 ',i,j,':',mop1[i,j],'  ');
229:                            writeln('B ',vet2[1],vet2[2]);
230:                            aux1[1] := vet2[1];
231:                            aux1[2] := vet2[2];
232:                            writeln('Copia 1: ',aux1[1],aux1[2]);
233:                          end;
234:                      3 : begin       //Matriz C 3x1
235:                            for i := 1 to 3 do
236:                                for j := 1 to 1 do
237:                                    mop1[i,j] := matC[i,j];
238:                            for i := 1 to 3 do
239:                                for j := 1 to 1 do
240:                                    write('mop1 ',i,j,':',mop1[i,j],'  ');
241:                            writeln('C ',vet3[1],vet3[2]);
242:                            aux1[1] := vet3[1];
243:                            aux1[2] := vet3[2];
244:                            writeln('Copia 1: ',aux1[1],aux1[2]);
245:                          end;
246:                      4 : begin       //Matriz D 1x3
```

```
                        for i := 1 to 1 do
                            for j := 1 to 3 do
                                mop1[i,j] := matD[i,j];
                        for i := 1 to 1 do
                            for j := 1 to 3 do
                                write('mop1 ',i,j,':',mop1[i,j],'  ');
                        writeln('D ',vet4[1],vet4[2]);
                        aux1[1] := vet4[1];
                        aux1[2] := vet4[2];
                        writeln('Copia 1: ',aux1[1],aux1[2]);
                    end;
                5 : begin       //Matriz E 3x4
                        for i := 1 to 3 do
                            for j := 1 to 4 do
                                mop1[i,j] := matE[i,j];
                        for i := 1 to 3 do
                            for j := 1 to 4 do
                                write('mop1 ',i,j,':',mop1[i,j],'  ');
                        writeln('E ',vet5[1],vet5[2]);
                        aux1[1] := vet5[1];
                        aux1[2] := vet5[2];
                        writeln('Copia 1: ',aux1[1],aux1[2]);
                    end;
                6 : begin       //Matriz F 4x2
                        for i := 1 to 4 do
                            for j := 1 to 2 do
                                mop1[i,j] := matF[i,j];
                        for i := 1 to 4 do
                            for j := 1 to 2 do
                                write('mop1 ',i,j,':',mop1[i,j],'  ');
                        writeln('F ',vet6[1],vet6[2]);
                        aux1[1] := vet6[1];
                        aux1[2] := vet6[2];
                        writeln('Copia 1: ',aux1[1],aux1[2]);
                    end;
                7 : begin       //Matriz G 3x3
                        for i := 1 to 3 do
                            for j := 1 to 3 do
                                mop1[i,j] := matG[i,j];
                        for i := 1 to 3 do
                            for j := 1 to 3 do
                                write('mop1 ',i,j,':',mop1[i,j],'  ');
                        writeln('G ',vet7[1],vet7[2]);
                        aux1[1] := vet7[1];
                        aux1[2] := vet7[2];
                        writeln('Copia 1: ',aux1[1],aux1[2]);
                    end
                else
                    begin
                        writeln('Opcao invalida!');
                        readkey;
                        clrscr;
                    end;
            end;
            write('Escolha a segunda matriz: ');       // 2° operando
            read(op2);
            case op2 of
                1 : begin      //Matriz A 3x3
                        for i := 1 to 3 do
                            for j := 1 to 3 do
                                mop2[i,j] := matA[i,j];
                        for i := 1 to 3 do
                            for j := 1 to 3 do
```

```
310:                             write('mop2 ',i,j,':',mop2[i,j],'  ');
311:                       writeln('A ',vet1[1],vet1[2]);
312:                       aux2[1] := vet1[1];
313:                       aux2[2] := vet1[2];
314:                       writeln('Copia 2: ',aux2[1],aux2[2]);
315:                     end;
316:                   2 : begin      //Matriz B 3x4
317:                       for i := 1 to 3 do
318:                           for j := 1 to 4 do
319:                               mop2[i,j] := matB[i,j];
320:                       for i := 1 to 3 do
321:                           for j := 1 to 4 do
322:                               write('mop2 ',i,j,':',mop2[i,j],'  ');
323:                       writeln('B ',vet2[1],vet2[2]);
324:                       aux2[1] := vet2[1];
325:                       aux2[2] := vet2[2];
326:                       writeln('Copia 2: ',aux2[1],aux2[2]);
327:                     end;
328:                   3 : begin      //Matriz C 3x1
329:                       for i := 1 to 3 do
330:                           for j := 1 to 1 do
331:                               mop2[i,j] := matC[i,j];
332:                       for i := 1 to 3 do
333:                           for j := 1 to 1 do
334:                               write('mop2 ',i,j,':',mop2[i,j],'  ');
335:                       writeln('C ',vet3[1],vet3[2]);
336:                       aux2[1] := vet3[1];
337:                       aux2[2] := vet3[2];
338:                       writeln('Copia 2: ',aux2[1],aux2[2]);
339:                     end;
340:                   4 : begin      //Matriz D 1x3
341:                       for i := 1 to 1 do
342:                           for j := 1 to 3 do
343:                               mop2[i,j] := matD[i,j];
344:                       for i := 1 to 1 do
345:                           for j := 1 to 3 do
346:                               write('mop2 ',i,j,':',mop2[i,j],'  ');
347:                       writeln('D ',vet4[1],vet4[2]);
348:                       aux2[1] := vet4[1];
349:                       aux2[2] := vet4[2];
350:                       writeln('Copia 2: ',aux2[1],aux2[2]);
351:                     end;
352:                   5 : begin      //Matriz E 3x4
353:                       for i := 1 to 3 do
354:                           for j := 1 to 4 do
355:                               mop2[i,j] := matE[i,j];
356:                       for i := 1 to 3 do
357:                           for j := 1 to 4 do
358:                               write('mop2 ',i,j,':',mop2[i,j],'  ');
359:                       writeln('E ',vet5[1],vet5[2]);
360:                       aux2[1] := vet5[1];
361:                       aux2[2] := vet5[2];
362:                       writeln('Copia 2: ',aux2[1],aux2[2]);
363:                     end;
364:                   6 : begin      //Matriz F 4x2
365:                       for i := 1 to 4 do
366:                           for j := 1 to 2 do
367:                               mop2[i,j] := matF[i,j];
368:                       for i := 1 to 4 do
369:                           for j := 1 to 2 do
370:                               write('mop2 ',i,j,':',mop2[i,j],'  ');
371:                       writeln('F ',vet6[1],vet6[2]);
372:                       aux2[1] := vet6[1];
```

```
373:                          aux2[2] := vet6[2];
374:                          writeln('Copia 2: ',aux2[1],aux2[2]);
375:                        end;
376:                    7 : begin      //Matriz G 3x3
377:                          for i := 1 to 3 do
378:                              for j := 1 to 3 do
379:                                  mop2[i,j] := matG[i,j];
380:                          for i := 1 to 3 do
381:                              for j := 1 to 3 do
382:                                  write('mop2 ',i,j,':',mop2[i,j],'  ');
383:                          writeln('G ',vet7[1],vet7[2]);
384:                          aux2[1] := vet7[1];
385:                          aux2[2] := vet7[2];
386:                          writeln('Copia 2: ',aux2[1],aux2[2]);
387:                        end
388:                    else
389:                        begin
390:                          writeln('Opcao invalida!');
391:                          readkey;
392:                          clrscr;
393:                        end;
394:                end;
395:                if(aux1[1] = aux2[1]) and (aux1[2] = aux2[2]) then     // Estrutura
    condicional
396:                   begin
397:                   writeln('Adicao de matrizes');                      // Operação de adição
398:                   for i := 1 to aux1[1] do
399:                       for j := 1 to aux1[2] do
400:                           matS[i,j] := mop1[i,j] + mop2[i,j];
401:                   for i := 1 to aux1[1] do
402:                       for j := 1 to aux2[1] do
403:                           writeln('Matriz soma = ',matS[i,j],' ');
404:                   writeln('Fim da operacao');
405:                   readkey;
406:                   clrscr;
407:                   end
408:                else
409:                   begin
410:                     writeln('Operacao impossivel devido a ordem das matrizes');
411:                     writeln('Deseja continuar? (y/n)');
412:                     continue := readkey;
413:                     clrscr;
414:                   end;
415:             end;                                                  // Fim - Adição
416:          2 : begin
417:                writeln('Subtracao de matrizes');                  // Subtração de
    matrizes
418:                writeln;
419:                write('Escolha a primeira matriz: ');
420:                read(op1);                                         // 1° operando
421:                case op1 of
422:                    1 : begin       //Matriz A 3x3
423:                          for i := 1 to 3 do
424:                              for j := 1 to 3 do
425:                                  mop1[i,j] := matA[i,j];
426:                          for i := 1 to 3 do
427:                              for j := 1 to 3 do
428:                                  write('mop1 ',i,j,':',mop1[i,j],'  ');
429:                          writeln('A ',vet1[1],vet1[2]);
430:                          aux1[1] := vet1[1];
431:                          aux1[2] := vet1[2];
432:                          writeln('Copia 1: ',aux1[1],aux1[2]);
433:                        end;
```

```
434:                    2 : begin       //Matriz B 3x4
435:                          for i := 1 to 3 do
436:                              for j := 1 to 4 do
437:                                  mop1[i,j] := matB[i,j];
438:                          for i := 1 to 3 do
439:                              for j := 1 to 4 do
440:                                  write('mop1 ',i,j,':',mop1[i,j],'  ');
441:                          writeln('B ',vet2[1],vet2[2]);
442:                          aux1[1] := vet2[1];
443:                          aux1[2] := vet2[2];
444:                          writeln('Copia 1: ',aux1[1],aux1[2]);
445:                        end;
446:                    3 : begin       //Matriz C 3x1
447:                          for i := 1 to 3 do
448:                              for j := 1 to 1 do
449:                                  mop1[i,j] := matC[i,j];
450:                          for i := 1 to 3 do
451:                              for j := 1 to 1 do
452:                                  write('mop1 ',i,j,':',mop1[i,j],'  ');
453:                          writeln('C ',vet3[1],vet3[2]);
454:                          aux1[1] := vet3[1];
455:                          aux1[2] := vet3[2];
456:                          writeln('Copia 1: ',aux1[1],aux1[2]);
457:                        end;
458:                    4 : begin       //Matriz D 1x3
459:                          for i := 1 to 1 do
460:                              for j := 1 to 3 do
461:                                  mop1[i,j] := matD[i,j];
462:                          for i := 1 to 1 do
463:                              for j := 1 to 3 do
464:                                  write('mop1 ',i,j,':',mop1[i,j],'  ');
465:                          writeln('D ',vet4[1],vet4[2]);
466:                          aux1[1] := vet4[1];
467:                          aux1[2] := vet4[2];
468:                          writeln('Copia 1: ',aux1[1],aux1[2]);
469:                        end;
470:                    5 : begin       //Matriz E 3x4
471:                          for i := 1 to 3 do
472:                              for j := 1 to 4 do
473:                                  mop1[i,j] := matE[i,j];
474:                          for i := 1 to 3 do
475:                              for j := 1 to 4 do
476:                                  write('mop1 ',i,j,':',mop1[i,j],'  ');
477:                          writeln('E ',vet5[1],vet5[2]);
478:                          aux1[1] := vet5[1];
479:                          aux1[2] := vet5[2];
480:                          writeln('Copia 1: ',aux1[1],aux1[2]);
481:                        end;
482:                    6 : begin       //Matriz F 4x2
483:                          for i := 1 to 4 do
484:                              for j := 1 to 2 do
485:                                  mop1[i,j] := matF[i,j];
486:                          for i := 1 to 4 do
487:                              for j := 1 to 2 do
488:                                  write('mop1 ',i,j,':',mop1[i,j],'  ');
489:                          writeln('F ',vet6[1],vet6[2]);
490:                          aux1[1] := vet6[1];
491:                          aux1[2] := vet6[2];
492:                          writeln('Copia 1: ',aux1[1],aux1[2]);
493:                        end;
494:                    7 : begin       //Matriz G 3x3
495:                          for i := 1 to 3 do
496:                              for j := 1 to 3 do
```

```
                                mop1[i,j] := matG[i,j];
                            for i := 1 to 3 do
                                for j := 1 to 3 do
                                    write('mop1 ',i,j,':',mop1[i,j],'  ');
                            writeln('G ',vet7[1],vet7[2]);
                            aux1[1] := vet7[1];
                            aux1[2] := vet7[2];
                            writeln('Copia 1: ',aux1[1],aux1[2]);
                        end
                    else
                        begin
                          writeln('Opcao invalida!');
                          readkey;
                          clrscr;
                        end;
                end;
                write('Escolha a segunda matriz: ');              // 2° operando
                read(op2);
                case op2 of
                    1 : begin      //Matriz A 3x3
                          for i := 1 to 3 do
                              for j := 1 to 3 do
                                  mop2[i,j] := matA[i,j];
                          for i := 1 to 3 do
                              for j := 1 to 3 do
                                  write('mop2 ',i,j,':',mop2[i,j],'  ');
                          writeln('A ',vet1[1],vet1[2]);
                          aux2[1] := vet1[1];
                          aux2[2] := vet1[2];
                          writeln('Copia 2: ',aux2[1],aux2[2]);
                        end;
                    2 : begin      //Matriz B 3x4
                          for i := 1 to 3 do
                              for j := 1 to 4 do
                                  mop2[i,j] := matB[i,j];
                          for i := 1 to 3 do
                              for j := 1 to 4 do
                                  write('mop2 ',i,j,':',mop2[i,j],'  ');
                          writeln('B ',vet2[1],vet2[2]);
                          aux2[1] := vet2[1];
                          aux2[2] := vet2[2];
                          writeln('Copia 2: ',aux2[1],aux2[2]);
                        end;
                    3 : begin      //Matriz C 3x1
                          for i := 1 to 3 do
                              for j := 1 to 1 do
                                  mop2[i,j] := matC[i,j];
                          for i := 1 to 3 do
                              for j := 1 to 1 do
                                  write('mop2 ',i,j,':',mop2[i,j],'  ');
                          writeln('C ',vet3[1],vet3[2]);
                          aux2[1] := vet3[1];
                          aux2[2] := vet3[2];
                          writeln('Copia 2: ',aux2[1],aux2[2]);
                        end;
                    4 : begin      //Matriz D 1x3
                          for i := 1 to 1 do
                              for j := 1 to 3 do
                                  mop2[i,j] := matD[i,j];
                          for i := 1 to 1 do
                              for j := 1 to 3 do
                                  write('mop2 ',i,j,':',mop2[i,j],'  ');
                          writeln('D ',vet4[1],vet4[2]);
```

```
560:                           aux2[1] := vet4[1];
561:                           aux2[2] := vet4[2];
562:                           writeln('Copia 2: ',aux2[1],aux2[2]);
563:                         end;
564:                       5 : begin      //Matriz E 3x4
565:                           for i := 1 to 3 do
566:                               for j := 1 to 4 do
567:                                   mop2[i,j] := matE[i,j];
568:                           for i := 1 to 3 do
569:                               for j := 1 to 4 do
570:                                   write('mop2 ',i,j,':',mop2[i,j],'  ');
571:                           writeln('E ',vet5[1],vet5[2]);
572:                           aux2[1] := vet5[1];
573:                           aux2[2] := vet5[2];
574:                           writeln('Copia 2: ',aux2[1],aux2[2]);
575:                         end;
576:                       6 : begin      //Matriz F 4x2
577:                           for i := 1 to 4 do
578:                               for j := 1 to 2 do
579:                                   mop2[i,j] := matF[i,j];
580:                           for i := 1 to 4 do
581:                               for j := 1 to 2 do
582:                                   write('mop2 ',i,j,':',mop2[i,j],'  ');
583:                           writeln('F ',vet6[1],vet6[2]);
584:                           aux2[1] := vet6[1];
585:                           aux2[2] := vet6[2];
586:                           writeln('Copia 2: ',aux2[1],aux2[2]);
587:                         end;
588:                       7 : begin      //Matriz G 3x3
589:                           for i := 1 to 3 do
590:                               for j := 1 to 3 do
591:                                   mop2[i,j] := matG[i,j];
592:                           for i := 1 to 3 do
593:                               for j := 1 to 3 do
594:                                   write('mop2 ',i,j,':',mop2[i,j],'  ');
595:                           writeln('G ',vet7[1],vet7[2]);
596:                           aux2[1] := vet7[1];
597:                           aux2[2] := vet7[2];
598:                           writeln('Copia 2: ',aux2[1],aux2[2]);
599:                         end
600:                       else
601:                           begin
602:                           writeln('Opcao invalida!');
603:                           readkey;
604:                           clrscr;
605:                         end;
606:
607:                   end;
608:                   if(aux1[1] = aux2[1]) and (aux1[2] = aux2[2]) then      // Estrutura
    condicional
609:                     begin
610:                     writeln('Subtracao de matrizes');                     // Operação de
    subtração
611:                     for i := 1 to aux1[1] do
612:                         for j := 1 to aux1[2] do
613:                             matSub[i,j] := mop1[i,j] - mop2[i,j];
614:                     for i := 1 to aux1[1] do
615:                         for j := 1 to aux2[1] do
616:                             writeln('Matriz diferenca = ',matSub[i,j],' ');
617:                     writeln('Fim da operacao');
618:                     readkey;
619:                     clrscr;
620:                     end
```

```
621:                 else
622:                    begin
623:                      writeln('Operacao impossivel devido a ordem das matrizes');
624:                      writeln('Deseja continuar? (y/n)');
625:                      continue := readkey;
626:                      clrscr;
627:                    end;                                          // Fim - Subtração
628:               end;
629:            3 : begin                                             // Multiplicação por
    constante
630:                 writeln('Multiplicacao por valor constante');
631:                 writeln;
632:                 write('Escolha uma matriz: ');
633:                 read(op1);                                       // Matriz - escolha
634:                 case op1 of
635:                      1 : begin       //Matriz A 3x3
636:                            for i := 1 to 3 do
637:                                for j := 1 to 3 do
638:                                    mop1[i,j] := matA[i,j];
639:                            for i := 1 to 3 do
640:                                for j := 1 to 3 do
641:                                    write('mop1 ',i,j,':',mop1[i,j],'  ');
642:                            writeln('A ',vet1[1],vet1[2]);
643:                            aux1[1] := vet1[1];
644:                            aux1[2] := vet1[2];
645:                            writeln('Copia 1: ',aux1[1],aux1[2]);
646:                          end;
647:                      2 : begin       //Matriz B 3x4
648:                            for i := 1 to 3 do
649:                                for j := 1 to 4 do
650:                                    mop1[i,j] := matB[i,j];
651:                            for i := 1 to 3 do
652:                                for j := 1 to 4 do
653:                                    write('mop1 ',i,j,':',mop1[i,j],'  ');
654:                            writeln('B ',vet2[1],vet2[2]);
655:                            aux1[1] := vet2[1];
656:                            aux1[2] := vet2[2];
657:                            writeln('Copia 1: ',aux1[1],aux1[2]);
658:                          end;
659:                      3 : begin       //Matriz C 3x1
660:                            for i := 1 to 3 do
661:                                for j := 1 to 1 do
662:                                    mop1[i,j] := matC[i,j];
663:                            for i := 1 to 3 do
664:                                for j := 1 to 1 do
665:                                    write('mop1 ',i,j,':',mop1[i,j],'  ');
666:                            writeln('C ',vet3[1],vet3[2]);
667:                            aux1[1] := vet3[1];
668:                            aux1[2] := vet3[2];
669:                            writeln('Copia 1: ',aux1[1],aux1[2]);
670:                          end;
671:                      4 : begin       //Matriz D 1x3
672:                            for i := 1 to 1 do
673:                                for j := 1 to 3 do
674:                                    mop1[i,j] := matD[i,j];
675:                            for i := 1 to 1 do
676:                                for j := 1 to 3 do
677:                                    write('mop1 ',i,j,':',mop1[i,j],'  ');
678:                            writeln('D ',vet4[1],vet4[2]);
679:                            aux1[1] := vet4[1];
680:                            aux1[2] := vet4[2];
681:                            writeln('Copia 1: ',aux1[1],aux1[2]);
682:                          end;
```

```
683:                      5 : begin       //Matriz E 3x4
684:                            for i := 1 to 3 do
685:                                for j := 1 to 4 do
686:                                    mop1[i,j] := matE[i,j];
687:                            for i := 1 to 3 do
688:                                for j := 1 to 4 do
689:                                    write('mop1 ',i,j,':',mop1[i,j],'  ');
690:                            writeln('E ',vet5[1],vet5[2]);
691:                            aux1[1] := vet5[1];
692:                            aux1[2] := vet5[2];
693:                            writeln('Copia 1: ',aux1[1],aux1[2]);
694:                          end;
695:                      6 : begin       //Matriz F 4x2
696:                            for i := 1 to 4 do
697:                                for j := 1 to 2 do
698:                                    mop1[i,j] := matF[i,j];
699:                            for i := 1 to 4 do
700:                                for j := 1 to 2 do
701:                                    write('mop1 ',i,j,':',mop1[i,j],'  ');
702:                            writeln('F ',vet6[1],vet6[2]);
703:                            aux1[1] := vet6[1];
704:                            aux1[2] := vet6[2];
705:                            writeln('Copia 1: ',aux1[1],aux1[2]);
706:                          end;
707:                      7 : begin       //Matriz G 3x3
708:                            for i := 1 to 3 do
709:                                for j := 1 to 3 do
710:                                    mop1[i,j] := matG[i,j];
711:                            for i := 1 to 3 do
712:                                for j := 1 to 3 do
713:                                    write('mop1 ',i,j,':',mop1[i,j],'  ');
714:                            writeln('G ',vet7[1],vet7[2]);
715:                            aux1[1] := vet7[1];
716:                            aux1[2] := vet7[2];
717:                            writeln('Copia 1: ',aux1[1],aux1[2]);
718:                          end
719:                      else
720:                          begin
721:                              writeln('Opcao invalida!');
722:                              readkey;
723:                              clrscr;
724:                          end;
725:                end;
726:                write('Digite o valor real: ');                 // Constante numérica
727:                read(vr);
728:                writeln;
729:                writeln('Multiplicacao de matrizes por valores constantes');    //
    Multiplicação por
730:                for i := 1 to aux1[1] do                                        // constante
731:                    for j := 1 to aux1[2] do
732:                        matm2k[i,j] := mop1[i,j] * vr;
733:                for i := 1 to aux1[1] do
734:                    for j := 1 to aux1[2] do
735:                        writeln('Matriz produto por ',vr,' ','(',i,j,')',' = ',matm2k[i,j],'
     ');
736:                writeln('Fim da operacao');
737:                readkey;
738:                clrscr;
739:              end;                                                  // Fim
740:           4 : begin
741:                writeln('Multiplicacao de matrizes');              // Multiplicação M2M
742:                write('Escolha a primeira matriz: ');
743:                read(op1);
```

```
744:                  case op1 of                                    // 1° operando
745:                       1 : begin       //Matriz A 3x3
746:                             for i := 1 to 3 do
747:                                 for j := 1 to 3 do
748:                                     mop1[i,j] := matA[i,j];
749:                             for i := 1 to 3 do
750:                                 for j := 1 to 3 do
751:                                     write('mop1 ',i,j,':',mop1[i,j],'  ');
752:                             writeln('A ',vet1[1],vet1[2]);
753:                             aux1[1] := vet1[1];
754:                             aux1[2] := vet1[2];
755:                             writeln('Copia 1: ',aux1[1],aux1[2]);
756:                           end;
757:                       2 : begin       //Matriz B 3x4
758:                             for i := 1 to 3 do
759:                                 for j := 1 to 4 do
760:                                     mop1[i,j] := matB[i,j];
761:                             for i := 1 to 3 do
762:                                 for j := 1 to 4 do
763:                                     write('mop1 ',i,j,':',mop1[i,j],'  ');
764:                             writeln('B ',vet2[1],vet2[2]);
765:                             aux1[1] := vet2[1];
766:                             aux1[2] := vet2[2];
767:                             writeln('Copia 1: ',aux1[1],aux1[2]);
768:                           end;
769:                       3 : begin       //Matriz C 3x1
770:                             for i := 1 to 3 do
771:                                 for j := 1 to 1 do
772:                                     mop1[i,j] := matC[i,j];
773:                             for i := 1 to 3 do
774:                                 for j := 1 to 1 do
775:                                     write('mop1 ',i,j,':',mop1[i,j],'  ');
776:                             writeln('C ',vet3[1],vet3[2]);
777:                             aux1[1] := vet3[1];
778:                             aux1[2] := vet3[2];
779:                             writeln('Copia 1: ',aux1[1],aux1[2]);
780:                           end;
781:                       4 : begin       //Matriz D 1x3
782:                             for i := 1 to 1 do
783:                                 for j := 1 to 3 do
784:                                     mop1[i,j] := matD[i,j];
785:                             for i := 1 to 1 do
786:                                 for j := 1 to 3 do
787:                                     write('mop1 ',i,j,':',mop1[i,j],'  ');
788:                             writeln('D ',vet4[1],vet4[2]);
789:                             aux1[1] := vet4[1];
790:                             aux1[2] := vet4[2];
791:                             writeln('Copia 1: ',aux1[1],aux1[2]);
792:                           end;
793:                       5 : begin       //Matriz E 3x4
794:                             for i := 1 to 3 do
795:                                 for j := 1 to 4 do
796:                                     mop1[i,j] := matE[i,j];
797:                             for i := 1 to 3 do
798:                                 for j := 1 to 4 do
799:                                     write('mop1 ',i,j,':',mop1[i,j],'  ');
800:                             writeln('E ',vet5[1],vet5[2]);
801:                             aux1[1] := vet5[1];
802:                             aux1[2] := vet5[2];
803:                             writeln('Copia 1: ',aux1[1],aux1[2]);
804:                           end;
805:                       6 : begin       //Matriz F 4x2
806:                             for i := 1 to 4 do
```

```
                                for j := 1 to 2 do
                                    mop1[i,j] := matF[i,j];
                              for i := 1 to 4 do
                                  for j := 1 to 2 do
                                      write('mop1 ',i,j,':',mop1[i,j],'  ');
                              writeln('F ',vet6[1],vet6[2]);
                              aux1[1] := vet6[1];
                              aux1[2] := vet6[2];
                              writeln('Copia 1: ',aux1[1],aux1[2]);
                            end;
                       7 : begin       //Matriz G 3x3
                              for i := 1 to 3 do
                                  for j := 1 to 3 do
                                      mop1[i,j] := matG[i,j];
                              for i := 1 to 3 do
                                  for j := 1 to 3 do
                                      write('mop1 ',i,j,':',mop1[i,j],'  ');
                              writeln('G ',vet7[1],vet7[2]);
                              aux1[1] := vet7[1];
                              aux1[2] := vet7[2];
                              writeln('Copia 1: ',aux1[1],aux1[2]);
                            end
                       else
                           begin
                              writeln('Opcao invalida!');
                              readkey;
                              clrscr;
                           end;
                  end;
                  write('Escolha a segunda matriz: ');         // 2° operando
                  read(op2);
                  case op2 of
                       1 : begin      //Matriz A 3x3
                              for i := 1 to 3 do
                                  for j := 1 to 3 do
                                      mop2[i,j] := matA[i,j];
                              for i := 1 to 3 do
                                  for j := 1 to 3 do
                                      write('mop2 ',i,j,':',mop2[i,j],'  ');
                              writeln('A ',vet1[1],vet1[2]);
                              aux2[1] := vet1[1];
                              aux2[2] := vet1[2];
                              writeln('Copia 2: ',aux2[1],aux2[2]);
                            end;
                       2 : begin      //Matriz B 3x4
                              for i := 1 to 3 do
                                  for j := 1 to 4 do
                                      mop2[i,j] := matB[i,j];
                              for i := 1 to 3 do
                                  for j := 1 to 4 do
                                      write('mop2 ',i,j,':',mop2[i,j],'  ');
                              writeln('B ',vet2[1],vet2[2]);
                              aux2[1] := vet2[1];
                              aux2[2] := vet2[2];
                              writeln('Copia 2: ',aux2[1],aux2[2]);
                            end;
                       3 : begin      //Matriz C 3x1
                              for i := 1 to 3 do
                                  for j := 1 to 1 do
                                      mop2[i,j] := matC[i,j];
                              for i := 1 to 3 do
                                  for j := 1 to 1 do
                                      write('mop2 ',i,j,':',mop2[i,j],'  ');
```

```
870:                          writeln('C ',vet3[1],vet3[2]);
871:                          aux2[1] := vet3[1];
872:                          aux2[2] := vet3[2];
873:                          writeln('Copia 2: ',aux2[1],aux2[2]);
874:                      end;
875:                  4 : begin      //Matriz D 1x3
876:                          for i := 1 to 1 do
877:                              for j := 1 to 3 do
878:                                  mop2[i,j] := matD[i,j];
879:                          for i := 1 to 1 do
880:                              for j := 1 to 3 do
881:                                  write('mop2 ',i,j,':',mop2[i,j],'  ');
882:                          writeln('D ',vet4[1],vet4[2]);
883:                          aux2[1] := vet4[1];
884:                          aux2[2] := vet4[2];
885:                          writeln('Copia 2: ',aux2[1],aux2[2]);
886:                      end;
887:                  5 : begin      //Matriz E 3x4
888:                          for i := 1 to 3 do
889:                              for j := 1 to 4 do
890:                                  mop2[i,j] := matE[i,j];
891:                          for i := 1 to 3 do
892:                              for j := 1 to 4 do
893:                                  write('mop2 ',i,j,':',mop2[i,j],'  ');
894:                          writeln('E ',vet5[1],vet5[2]);
895:                          aux2[1] := vet5[1];
896:                          aux2[2] := vet5[2];
897:                          writeln('Copia 2: ',aux2[1],aux2[2]);
898:                      end;
899:                  6 : begin      //Matriz F 4x2
900:                          for i := 1 to 4 do
901:                              for j := 1 to 2 do
902:                                  mop2[i,j] := matF[i,j];
903:                          for i := 1 to 4 do
904:                              for j := 1 to 2 do
905:                                  write('mop2 ',i,j,':',mop2[i,j],'  ');
906:                          writeln('F ',vet6[1],vet6[2]);
907:                          aux2[1] := vet6[1];
908:                          aux2[2] := vet6[2];
909:                          writeln('Copia 2: ',aux2[1],aux2[2]);
910:                      end;
911:                  7 : begin      //Matriz G 3x3
912:                          for i := 1 to 3 do
913:                              for j := 1 to 3 do
914:                                  mop2[i,j] := matG[i,j];
915:                          for i := 1 to 3 do
916:                              for j := 1 to 3 do
917:                                  write('mop2 ',i,j,':',mop2[i,j],'  ');
918:                          writeln('G ',vet7[1],vet7[2]);
919:                          aux2[1] := vet7[1];
920:                          aux2[2] := vet7[2];
921:                          writeln('Copia 2: ',aux2[1],aux2[2]);
922:                      end
923:                  else
924:                      begin
925:                         writeln('Opcao invalida!');
926:                         readkey;
927:                         clrscr;
928:                      end;
929:              end;
930:              readkey;
931:              clrscr;            //Limpa a tela
932:              if(aux1[2] = aux2[1]) then              // Estrutura condicional
```

```
933:                    begin
934:                    matP[1,1] := 0;
935:                    matP[1,2] := 0;                         // Zerando a matriz
936:                    matP[1,3] := 0;                         // produto - matP
937:                    matP[1,4] := 0;
938:                    matP[1,5] := 0;
939:                    matP[2,1] := 0;
940:                    matP[2,2] := 0;
941:                    matP[2,3] := 0;
942:                    matP[2,4] := 0;
943:                    matP[2,5] := 0;
944:                    matP[3,1] := 0;
945:                    matP[3,2] := 0;
946:                    matP[3,3] := 0;
947:                    matP[3,4] := 0;
948:                    matP[3,5] := 0;
949:                    matP[4,1] := 0;
950:                    matP[4,2] := 0;
951:                    matP[4,3] := 0;
952:                    matP[4,4] := 0;
953:                    matP[4,5] := 0;
954:                    matP[5,1] := 0;
955:                    matP[5,2] := 0;
956:                    matP[5,3] := 0;
957:                    matP[5,4] := 0;
958:                    matP[5,5] := 0;
959:                    writeln('Multiplicacao de matrizes');     // Operação M2M
960:                    for i := 1 to aux1[1] do
961:                        for j := 1 to aux2[2] do
962:                            for k := 1 to aux1[1] do
963:                                matP[i,j] := mop1[i,k] * mop2[k,j] + matP[i,j];
964:                    for i := 1 to aux1[1] do
965:                        for j := 1 to aux2[2] do
966:                            writeln('Matriz produto ',i,'x',j,' = ',matP[i,j],' ');
967:                    writeln('Fim da operacao');
968:                    readkey;
969:                    clrscr;
970:                    end
971:                  else
972:                    begin
973:                      writeln('Impossivel, pois o numero de colunas da primeira matriz');
974:                      writeln('nao eh igual ao numero de linhas da segunda matriz!');
975:                      writeln('Deseja continuar? (y/n)');
976:                      continue := readkey;
977:                      clrscr;
978:                    end;
979:                end;                                        // Fim - M2M
980:            5 : begin                                       // Sair ou continuar
981:                  clrscr;
982:                  writeln('Sair');
983:                  writeln('Deseja continuar? (y/n)');
984:                  continue := readkey;
985:                  clrscr;
986:                end
987:            else
988:                begin
989:                  writeln ('Operacao invalida!!');
990:                  readkey;
991:                  clrscr;
992:                end
993:       end;
994:   end;
995:   readkey;
```

```
996: end.
997:
```