НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ФАКУЛЬТЕТ ІНФОРМАТИКИ І ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

**Лабораторна робота №1**

з дисципліни **«**Паралельні та розподілені обчислення**»**

Виконав:

студент 3 курсу гр. ІО-31

Долинний Олександр

№ ЗК 3110

Перевірив:

Корочкін О. В.

Київ 2015 р.

***Завдання:***

1.1 3.21 3.11

F1: A = sort(B) \* (MB \* MC)

F2: W = sort(R \* MT) \* (MX \* MS)

F3: R = sort(S + T) \* trans(MS \* MR)

***Лістинг програми:***

GNAT GPL 2015 (20150428-49)

Copyright 1992-2015, Free Software Foundation, Inc.

Compiling: lab1.adb

Source file time stamp: 2015-09-25 08:34:46

Compiled at: 2015-09-25 11:37:53

1. ----------------Main programm------------------------

2.

3. --Parallel and distributed computing.

4. --Labwork 1. Ada. Subprograms and packages

5. --Alexandr Dolinniy

6. --IO-31

7. --23.09.2015

8. --Func1: A = sort(B)\*(MB\*MC)

9. --Func2: W = sort(R\*MT)\*(MX\*MS)

10. --Func3: R = sort(S+T)\*trans(MS\*MR)

11.

12. with Data, Text\_IO, Ada.Integer\_Text\_IO;

13. use Text\_IO, Ada.Integer\_Text\_IO;

14.

15. procedure lab1 is

16. n: Integer := 3 ;

17. package data1 is new data (n);

18. use data1;

19. MB, MC, MT, MX, MS, MR: Matrix;

20. B, R, S, T, f1, f2, f3: Vector;

21. Begin

22.

23. --Filling matrices and vectors with ones

24. Matrix\_Filling\_Ones(MB);

25. Matrix\_Filling\_Ones(MC);

26. Matrix\_Filling\_Ones(MT);

27. Matrix\_Filling\_Ones(MX);

28. Matrix\_Filling\_Ones(MS);

29. Matrix\_Filling\_Ones(MR);

30. Vector\_Filling\_Ones(B);

31. Vector\_Filling\_Ones(R);

32. Vector\_Filling\_Ones(S);

33. Vector\_Filling\_Ones(T);

34.

35. --numberInMatrix(ma, 1, 2, 9);

36. --writeMatrix(mA);

37.

38. --Calculation functions

39. f1 := Func1(B, MB, MC);

40. f2 := Func2(R, MT, MX, MS);

41. f3 := Func3(S, T, MS, MR);

42.

43. --If the dimension of the vector N more than 7 result

44. -- function are not written on the screen

45. if n < 7 then

46. Put("Func1 = sort(B)\*(MB\*MC)");

47. Vector\_Output(f1);

48. New\_Line;

49. Put\_Line("Func2 = sort(R\*MT)\*(MX\*MS)");

50. Vector\_Output(f2);

51. New\_Line;

52. put("Func3 = sort(S+T)\*trans(MS\*MR)");

53. Vector\_Output(f3);

54. end if;

55. End lab1;

55 lines: No errors

GNAT GPL 2015 (20150428-49)

Copyright 1992-2015, Free Software Foundation, Inc.

Compiling: data.adb

Source file time stamp: 2015-09-25 08:34:46

Compiled at: 2015-09-25 11:38:16

1. -----------Package Data, body-----------

2.

3. with Text\_IO, Ada.Integer\_Text\_IO;

4. use Text\_IO, Ada.Integer\_Text\_IO;

5. package body Data is

6.

7. --Read Vector

8. procedure Vector\_Input(A: out Vector) is

9. begin

10. for i in 1..n loop

11. Get(A(i));

12. end loop;

13. end Vector\_Input;

14.

15. --Write vector on screen

16. procedure Vector\_Output(A: in Vector) is

17. begin

18. for i in 1..n loop

19. Put(A(i));

20. Put(" ");

21. end loop;

22. end Vector\_Output;

23.

24. --Read matrix

25. procedure Matrix\_Input(A: out Matrix) is

26. begin

27. for i in 1..n loop

28. for j in 1..n loop

29. Get(A(i)(j));

30. end loop;

31. end loop;

32. end Matrix\_Input;

33.

34. --Write matrix on screen

35. procedure Matrix\_Output (A: in Matrix) is

36. begin

37. for i in 1..n loop

38. for j in 1..n loop

39. Put(A(i)(j));

40. Put(" ");

41. end loop;

42. Put\_Line(" ");

43. end loop;

44. end Matrix\_Output;

45.

46. --Multiplication of matrices

47. function Matrix\_Multiplication(A, B: in Matrix) return Matrix is

48. P: Matrix;

49. S: Integer;

50. begin

51. for k in 1..n loop

52. for i in 1..n loop

53. s := 0;

54. for j in 1..n loop

55. S := S + A(k)(j)\*B(j)(i);

56. P(k)(i) := s;

57. end loop;

58. end loop;

59. end loop;

60. return P;

61. end Matrix\_Multiplication;

62.

63. --Multiplication of vector and matrix

64. function Vector\_Matrix\_Multiplication(A: in Vector; B: in Matrix) return Vector is

65. P: Vector;

66. s: Integer;

67. begin

68. for i in 1..n loop

69. s := 0;

70. for j in 1..n loop

71. S := s + A(i)\*B(j)(i);

72. end loop;

73. P(i) := S;

74. end loop;

75. return P;

76. end Vector\_Matrix\_Multiplication;

77.

78. --Sum of vectors

79. function Vector\_Sum(A, B: in Vector) return Vector is

80. S: Vector;

81. begin

82. for i in 1..n loop

83. S(i) := A(i)+B(i);

84. end loop;

85. return S;

86. end Vector\_Sum;

87.

88. --Sorting of vector

89. procedure Vector\_Sorting(A: in out Vector) is

90. S: Integer;

91. begin

92. for i in 1..n loop

93. for j in i..n loop

94. if A(i)>A(j) then

95. S:=A(j);

96. A(j):=A(i);

97. A(i):=S;

98. end if;

99. end loop;

100. end loop;

101. end Vector\_Sorting;

102.

103. --Transposition of Matrix

104. procedure Matrix\_Transposition(A: in out Matrix) is

105. S: Integer;

106. begin

107. for i in 1..n loop

108. for j in i..n loop

109. S:=A(j)(i);

110. A(j)(i):=A(i)(j);

111. A(i)(j):=S;

112. end loop;

113. end loop;

114. end Matrix\_Transposition;

115.

116. --Filling matrix with ones

117. procedure Matrix\_Filling\_Ones(A: out Matrix) is

118. begin

119. for i in 1..n loop

120. for j in 1..n loop

121. A(i)(j) := 1;

122. end loop;

123. end loop;

124. end Matrix\_Filling\_Ones;

125.

126. --Filling vector with ones

127. procedure Vector\_Filling\_Ones (A: out vector) is

128. begin

129. for i in 1..n loop

130. A(i) := 1;

131. end loop;

132. end Vector\_Filling\_Ones;

133.

134. procedure Matrix\_Filling\_Number(A: out Matrix; i,j,number:Integer) is

135. begin

136. A(i)(j) := number;

137. end Matrix\_Filling\_Number;

138.

139. procedure Vector\_Filling\_Number(A: out Vector; i,number:integer) is

140. begin

141. A(i) := number;

142. end Vector\_Filling\_Number;

143.

144.

145. --Calculation function 1

146. function Func1 (B: out Vector; MB, MC : in Matrix) return Vector is

147. MD:Matrix;

148. A:Vector;

149. begin

150. Vector\_Sorting(B);

151. MD:=Matrix\_Multiplication(MB,MC);

152. A:=Vector\_Matrix\_Multiplication(B,MD);

153. return A;

154. end Func1;

155.

156. --Calculation function 2

157. function Func2 (R: in Vector; MT, MX, MS : in Matrix) return Vector is

158. A,B:Vector;

159. MD: Matrix;

160. begin

161. A:=Vector\_Matrix\_Multiplication(R,MT);

162. Vector\_Sorting(A);

163. MD := Matrix\_Multiplication(MX, MS);

164. B:=Vector\_Matrix\_Multiplication(A,MD);

165. return B;

166. end Func2;

167.

168. --Calculation function 3

169. function Func3 (S,T: in Vector; MS, MR : in Matrix) return Vector is

170. R,A: Vector;

171. MD: Matrix;

172. begin

173. A := Vector\_Sum(S, T);

174. Vector\_Sorting(A);

175. MD := Matrix\_Multiplication(MS, MR);

176. Matrix\_Transposition(MD);

177. R:=Vector\_Matrix\_Multiplication(A,MD);

178. return R;

179. end Func3;

180. end Data;

Compiling: data.ads

Source file time stamp: 2015-09-25 08:34:46

Compiled at: 2015-09-25 11:38:16

1.

2. generic

3. n: Integer;

4. package data is

5.

6. ---Declaration of private types

7. type Vector is private;

8. type Matrix is private;

9.

10. --Read Vector

11. procedure Vector\_Input(A: out Vector);

12.

13. --Write vector on screen

14. procedure Vector\_Output(A: in Vector);

15.

16. --Read matrix

17. procedure Matrix\_Input(A: out Matrix);

18.

19. --Write matrix on screen

20. procedure Matrix\_Output (A: in Matrix);

21.

22. --Calculation function 1

23. function Func1 (B: out Vector; MB, MC : in Matrix) return Vector;

24.

25. --Calculation function 2

26. function Func2 (R: in Vector; MT, MX, MS : in Matrix) return Vector;

27.

28. --Calculation function 3

29. function Func3 (S,T: in Vector; MS, MR : in Matrix) return Vector;

30.

31.

32. --Filling matrix with ones

33. procedure Matrix\_Filling\_Ones(A: out Matrix);

34.

35. --Filling vector with ones

36. procedure Vector\_Filling\_Ones (A: out vector);

37.

38. procedure Matrix\_Filling\_Number(A: out Matrix; i,j,number:Integer);

39.

40. procedure Vector\_Filling\_Number(A: out Vector; i,number:integer);

41.

42.

43.

44. --Determination private types

45. private

46. type Vector is array (1..n) of Integer;

47. type Matrix is array (1..n) of Vector;

48.

49. end Data;

180 lines: No errors