НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №1

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

Виконав:

студент 3 курсу

ФІОТ гр. ІО-21

Кузьменко Володимир

Перевірив:

Корочкін О. В.

Київ 2014 р.

Технічне завдання:

F1: 1.24 E = A + C \* (MA \* MZ) + B;

F2: 2.14 MC = SORT (MA + MB \* MM);

F3: 3.18 p = MAX (SORT(MS) + MA \* MM)

Лістинг коду:

1. -----------------------------------------------------

2. -- --

3. -- Parallel and distributed computing --

4. -- Lab 1. Routines and packages --

5. -- --

6. -- --

7. -- F1: 1.24 E = A + C \* (MA \* MZ) + B --

8. -- F2: 2.14 MC = SORT(MA + MB \* MM) --

9. -- F3: 3.18 p = MAX(SORT(MS) + MA\* MM) --

10. -- --

11. -- Author Kuzmenko Vladimir Z. --

12. -- Group IO-21 --

13. -- Date 10.09.2014 --

14. -- --

15. -----------------------------------------------------

16.

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

18. with data;

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

20.

21. procedure Lab1 is

22. package data3 is new data (3);

23. use data3;

24.

25. p : Integer;

26. A, B, C, E : Vector;

27. MA, MB, MC, MM, MS, MZ : Matrix;

28. begin

29. --------------

30. --Input Data--

31. --------------

32.

33. Put\_Line ("Input vector A");

34. Vector\_Input (A);

35. Vector\_Output (A);

36. New\_Line;

37.

38. Put\_Line ("Input vector B");

39. Vector\_Input (B);

40. Vector\_Output (B);

41. New\_Line;

42.

43. Put\_Line ("Input vector C");

44. Vector\_Input (C);

45. Vector\_Output (C);

46. New\_Line;

47.

48. Put\_Line ("Input matrix MA");

49. Matrix\_Input (MA);

50. Matrix\_Output (MA);

51. New\_Line;

52.

53. Put\_Line ("Input matrix MB");

54. Matrix\_Input (MB);

55. Matrix\_Output (MB);

56. New\_Line;

57.

58. Put\_Line ("Input matrix MM");

59. Matrix\_Input (MM);

60. Matrix\_Output (MM);

61. New\_Line;

62.

63. Put\_Line ("Input matrix MS");

64. Matrix\_Input (MS);

65. Matrix\_Output (MS);

66. New\_Line;

67.

68. Put\_Line ("Input matrix MZ");

69. Matrix\_Input (MZ);

70. Matrix\_Output (MZ);

71. New\_Line;

72.

73. ---------------

74. --Calculation--

75. ---------------

76. Func1 (A, B, C, MA, MZ, E);

77. Func2 (MA, MB, MM, MC);

78. Func3 (MA, MM, MS, p);

79.

80. ---------------

81. --Output data--

82. ---------------

83. Put ("Function 1, vector E");

84. Vector\_Output (E);

85. New\_Line;

86.

87. Put ("Function 2, matrix MC");

88. Matrix\_Output (MC);

89. New\_Line;

90.

91. Put ("Function 3, value p");

92. Put (p);

93. New\_Line;

94.

95. ----------------------

96. --Use "private" type--

97. ----------------------

98.

99. --OK--

100. --appropriation--

101. B := A;

102. Put\_Line ("It's ok");

103. --compare--

104. if A = B then

105. Put\_Line ("It's ok");

106. end if;

107.

108. --error--

109. --B(1):=A(1)+2;

110.

111. end Lab1;

1.

2. ------------------------

3. -- File data.adb

4. -- group IO-21

5. -- Kuzmenko Vladimir Z.

6. --------------------------

7.

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

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

10. package body Data is

11.

12. --input vector

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

14. begin

15. for i in 1..n loop

16. Get(A(i));

17. end loop;

18. end Vector\_Input;

19.

20. --output vector

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

22. begin

23. for i in 1..n loop

24. Put(A(i));

25. end loop;

26. end Vector\_Output;

27.

28. -- input matrix

29. procedure Matrix\_Input (MA:out Matrix) is

30. begin

31. for i in 1..n loop

32. for j in 1..n loop

33. Get(MA(i)(j));

34. end loop;

35. end loop;

36. end Matrix\_Input;

37.

38. --output matrix

39. procedure Matrix\_Output (MA:in Matrix) is

40. begin

41. for i in 1..n loop

42. for j in 1..n loop

43. Put(MA(i)(j));

44. Put ("|");

45. end loop;

46. Put\_Line("");

47. end loop;

48. end Matrix\_Output;

49.

50. --multing matrix

51. function MultMatr (ma,mz:Matrix) return Matrix is

52. s:integer;

53. MT:Matrix;

54. begin

55. for i in 1..n loop

56. for j in 1..n loop

57. s:=0;

58. for k in 1..n loop

59. s:=s+ma(i)(k)\*mz(k)(j);

60. MT(i)(j):=s;

61. end loop;

62. end loop;

63. end loop;

64. return MT;

65. end MultMatr;

66.

67. --multing matrix on vector

68. function multMatrixOnVector(MA:Matrix; A:Vector) return Vector is

69. s:Integer;

70. R:Vector;

71. begin

72. for i in 1..n loop

73. s:=0;

74. for j in 1..n loop

75. s:=s+MA(i)(j)\*A(j);

76. end loop;

77. R(i):=s;

78. end loop;

79. return R;

80. end multMatrixOnVector;

81.

82. --addition vectors

83. function AddVec(B,C:Vector) return Vector is

84. K:Vector;

85. begin

86. for i in 1..n loop

87. K(i):=B(i)+C(i);

88. end loop;

89. return K;

90. end AddVec;

91.

92. --addition matrix

93. function AddMatr (ma,mb:Matrix) return Matrix is

94. m:Matrix;

95. begin

96. for i in 1..n loop

97. for j in 1..n loop

98. m(i)(j):=ma(i)(j)+mb(i)(j);

99. end loop;

100. end loop;

101. return m;

102. end AddMatr;

103.

104. --sorting vector

105. function SortVector(A:Vector) return Vector is

106. M:Vector:=A;

107. buf:Integer;

108. k:Integer;

109. begin

110. for i in 1..(n-1) loop

111. k:=i;

112. for j in (i+1)..n loop

113. if M(k)>M(j) then

114. k:=j;

115. end if;

116. end loop;

117. buf:=M(k);

118. M(k):=M(i);

119. M(i):=buf;

120. end loop;

121. return M;

122. end SortVector;

123.

124. -- sjrting matrix

125. function SortMatr(MA:Matrix) return Matrix is

126. MT:Matrix;

127. begin

128. for i in 1..n loop

129. MT(i):=sortVector(MA(i));

130. end loop;

131. return MT;

132. end SortMatr;

133.

134. --finding max element in matrix

135. function MaxElMatr (MD:Matrix) return Integer is

136. m:Integer:=MD(1)(1);

137. begin

138. for i in 1..n loop

139. for j in 1..n loop

140. if m<MD(i)(j) then

141. m:=MD(i)(j);

142. end if;

143. end loop;

144. end loop;

145. return m;

146. end MaxElMatr;

147.

148. --F1: 1.24 E = A + C \* (MA \* MZ) + B

149. procedure Func1(A,B,C:in Vector;MA,MZ: in Matrix; E:out Vector) is

150. begin

151. E:=AddVec(AddVec(multMatrixOnVector(MultMatr(MA,MZ),C),A),B);

152. end Func1;

153.

154. --F2: 2.14 MC = SORT(MA + MB \* MM)

155. procedure Func2(MA,MB,MM:in Matrix; MC:out Matrix) is

156. begin

157. MC:= SortMatr(AddMatr(MultMatr(MB,MM),MA));

158. end Func2;

159.

160. --F3: 3.18 p = MAX(SORT(MS) + MA\* MM)

161. procedure Func3(MA,MM,MS: in Matrix; p: out Integer) is

162. begin

163. p:= MaxElMatr(AddMatr(SortMatr(MS),MultMatr(MA,MM)));

164. end Func3;

165.

166. end Data;

167.

Compiling: data.ads (source file time stamp: 2014-09-09 20:54:28)

1.

2. ------------------------

3. -- File data.ads

4. -- group IO-21

5. -- Kuzmenko Vladimir Z.

6. --------------------------

7.

8. generic

9. n : Integer;

10. package data is

11.

12. type Vector is private;

13. type Matrix is private;

14.

15. -- input vector

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

17. -- output vector

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

19. -- input matrix

20. procedure Matrix\_Input (MA:out Matrix);

21. -- output matrix

22. procedure Matrix\_Output (MA:in Matrix);

23.

24. --F1: 1.24 E = A + C \* (MA \* MZ) + B

25. procedure Func1(A,B,C:in Vector;MA,MZ: in Matrix; E:out Vector);

26.

27. --F2: 2.14 MC = SORT(MA + MB \* MM)

28. procedure Func2(MA,MB,MM:in Matrix; MC:out Matrix);

29.

30. --F3: 3.18 p = MAX(SORT(MS) + MA\* MM)

31. procedure Func3(MA,MM,MS: in Matrix; p: out Integer);

32.

33. private

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

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

36.

37. end data;