Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Лабораторна робота №8**

*по курсу*

*«Паралельні та розподілені обчислення»  
OpenMP*

Виконав:

студент III курсу

гр. IO-82

Куцовол В.В.

Київ 2010

Lab8PRO.cpp

1 /\*\*

2 \* @author Kutsovol Victor IO-82

3 \* PRO lab 4

4 \* F1: 1.18 d=(A\*B)-(C\*B)

5 \* F2: 2.19 v=MAX(MA+MB\*MC)

6 \* F3: 3.20 R=SORT(B-C)\*SORT(MA\*MB)

7 \*/

8 #include "stdafx.h"

9 #include <windows.h>

10 #include <omp.h>

11 #include "MyMath.h"

12

13 void ThreadFunc1()

14 {

15 printf("Task1 Started \n");

16 int n = 500;

17 int m = 1;

18 int \*A = new int[n];

19 int \*B = new int[n];

20 int \*C = new int[n];

21 int d;

22 for (int i = 0; i < n; i++) {

23 A[i] = m;

24 B[i] = m;

25 C[i] = m;

26 }

27 d = func1(A,B,C,n);

28 printf("F1: %d ", d);

29 printf("\nTask1 Finished \n");

30 }

31 //-----------------------------------------

32 void ThreadFunc2()

33 {

34 printf("Task2 Started \n");

35 int n = 500;

36 int m = 1;

37 int \*\*MA = new int\*[n];

38 int \*\*MB = new int\*[n];

39 int \*\*MC = new int\*[n];

40 int v;

41 for (int i = 0; i < n; i++) {

42 MA[i] = new int[n];

43 MB[i] = new int[n];

44 MC[i] = new int[n];

45 }

46 for (int i = 0; i < n; i++) {

47 for (int j = 0; j < n; j++) {

48 MA[i][j] = m;

49 MB[i][j] = m;

50 MC[i][j] = m;

51 }

52 }

53 v = func2(MA, MB, MC, n);

54 printf("F2: %d ", v);

55 printf("\nTask2 Finished \n");

56 }

57 //-----------------------------------------

58 void ThreadFunc3()

59 {

60 printf("Task3 Started \n");

61 int n = 500;

62 int m = 1;

63 int \*B = new int[n];

64 int \*C = new int[n];

65 int \*R = new int[n];

66 int \*\*MA = new int\*[n];

67 int \*\*MB = new int\*[n];

68 for (int i = 0; i < n; i++) {

69 MA[i] = new int[n];

70 MB[i] = new int[n];

71 }

72 for (int i = 0; i < n; i++) {

73 B[i] = m;

74 C[i] = m;

75 for (int j = 0; j < n; j++) {

76 MA[i][j] = m;

77 MB[i][j] = m;

78 }

79 }

80 R = func3(B,C,MA,MB,n);

81 printf("F3:\n");

82 for (int i = 0; i < n; i++) {

83 printf("%d ", R[i]);

84 }

85 printf("\nTask3 Finished \n");

86 }

87 //-----------------------------------------

88 int main()

89 {

90 int thrdNumb=3;

91 omp\_set\_num\_threads(thrdNumb);

92 #pragma omp parallel

93 {

94 #pragma omp single nowait

95 ThreadFunc1();

96 #pragma omp single nowait

97 ThreadFunc2();

98 #pragma omp single nowait

99 ThreadFunc3();

100 }

101 return 0;

102}

MyMath.cpp

1 /\*\*

2 \* @author Kutsovol Victor IO-82

3 \* PRO lab 4

4 \* F1: 1.18 d=(A\*B)-(C\*B)

5 \* F2: 2.19 v=MAX(MA+MB\*MC)

6 \* F3: 3.20 R=SORT(B-C)\*SORT(MA\*MB)

7 \*/

8 #include "StdAfx.h"

9 #include "MyMath.h"

10

11 int multVonV(int\* A, int\* B, int n) {

12 int res = 0;

13 for (int i = 0; i < n; i++) {

14 res += A[i]\*B[i];

15 }

16 return res;

17 }

18 int\*\* add(int\*\* MA, int\*\* MB, int n) {

19 int\*\* res = new int\*[n];

20 for (int i = 0; i < n; i++) {

21 res[i] = new int[n];

22 }

23 for (int i = 0; i < n; i++)

24 for (int j = 0; j < n; j++) {

25 res[i][j] = MA[i][j] + MB[i][j];

26 }

27 return res;

28 }

29 int\*\* multMatrix(int\*\* MB, int\*\* MC, int n) {

30 int\*\* res = new int\*[n];

31 for (int i = 0; i < n; i++) {

32 res[i] = new int[n];

33 }

34 int s;

35 for (int i = 0; i < n; i++)

36 for (int j = 0; j < n; j++) {

37 s = 0;

38 for (int k = 0; k < n; k++)

39 s = s + MB[i][k] \* MC[k][j];

40 res[i][j] = s;

41 }

42 return res;

43 }

44

45 int maxElMatr(int\*\* MA, int n) {

46 int res = MA[0][0];

47 for (int i = 0; i < n; i++)

48 for (int j = 0; i < n; i++)

49 if (res < MA[i][j])

50 res = MA[i][j];

51 return res;

52 }

53

54 void sort(int\*\* m, int n) {

55 int nm = n\*n;

56 int buf;

57 for (int g = 0; g < nm; g++) {

58 for (int i = 0; i < n; i++) {

59 for (int j = 0; j < n; j++) {

60 if (j < n - 1) {

61 if (m[i][j] > m[i][j+1]) {

62 buf = m[i][j];

63 m[i][j] = m[i][j+1];

64 m[i][j+1] = buf;

65 }

66 } else {

67 if (i < n - 1) {

68 if (m[i][j] > m[i+1][0]) {

69 buf = m[i][j];

70 m[i][j] = m[i+1][0];

71 m[i+1][0] = buf;

72 }

73 }

74 }

75 }

76 }

77 }

78 }

79 int\* sub(int\* B, int\* C, int n) {

80 int\* res = new int[n];

81 for (int i = 0; i < n; i++)

82 res[i] = B[i] - C[i];

83 return res;

84 }

85

86 int\* multMonV(int\*\* MA, int\* A, int n) {

87 int\* res = new int[n];

88 int s;

89 for (int i = 0; i < n; i++) {

90 s = 0;

91 for (int j = 0; j < n; j++)

92 s += MA[i][j] \* A[j];

93 res[i] = s;

94 }

95 return res;

96 }

97

98 void sort(int\* A, int n){

99 for (int i = 0; i<n-1; ++i){

100 int tmp = 0;

101 for (int j=(i+1); j<n; ++j)

102 if (A[i]>A[j]) {

103 tmp = A[i];

104 A[i] = A[j];

105 A[j] = tmp;

106 }

107 }

108 }

109

110 int func1(int\* A, int\* B, int\* C, int n) {

111 int res;

112 res = multVonV(A,B,n) - multVonV(C,B,n);

113 return res;

114 }

115 int func2(int\*\* MA, int\*\* MB, int\*\* MC, int n) {

116 int res;

117 res = maxElMatr(add(MA,multMatrix(MB,MC,n),n),n);

118 return res;

119 }

120 int\* func3(int\* B, int\* C, int\*\* MA, int\*\* MB, int n) {

121 int\* res;

122 int\* subV = sub(B,C,n);

123 sort(subV,n);

124 int\*\* sortmulM = multMatrix(MA,MB,n);

125 sort (sortmulM,n);

126 res = multMonV(sortmulM,subV,n);

127 return res;

MyMath.h

1 /\*\*

2 \* @author Kutsovol Victor IO-82

3 \* PRO lab 4

4 \* F1: 1.18 d=(A\*B)-(C\*B)

5 \* F2: 2.19 v=MAX(MA+MB\*MC)

6 \* F3: 3.20 R=SORT(B-C)\*SORT(MA\*MB)

7 \*/

8 int func1(int\* A, int\* B, int\* C, int n);

9 int func2(int\*\* MA, int\*\* MB, int\*\* MC, int n);

10 int\* func3(int\* B, int\* C, int\*\* MA, int\*\* MB, int n);