Chapter11-程序源代码与运行结果

11.3

// Source Codes:

#include <stdio.h>

**void** inputNumber(**int** \*p, **const** **char**\* prompt) {

printf(prompt);

**while**(!scanf("%d", p)) {

printf(prompt);

**while**(getchar()!='\n');

}

**return**;

}

**int** main() {

**char** \*a[] = {**NULL**, "January", "February", "March", "April", "May", "June", "July", "August", "September", "October", "November", "December"};

**int** i=0;

inputNumber(&i, "Input Month # > ");

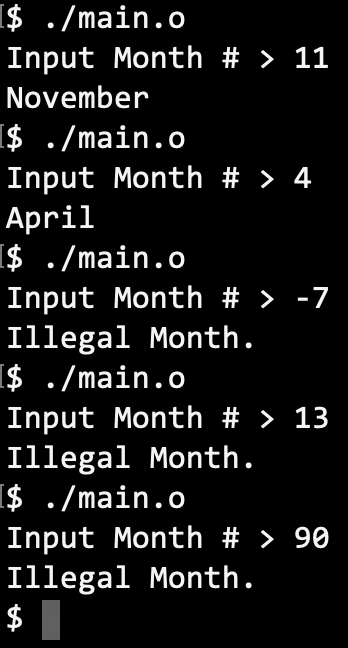
**if** (i<=12 && i>=1) puts(a[i]);

**else** puts("Illegal Month.");

**return** 0;

}

// Results:



11.4

// Source Codes:

#include <stdio.h>

#define N 12

**void** inputNumber(**int** \*p, **const** **char**\* prompt) {

printf(prompt);

**while**(!scanf("%d", p)) {

printf(prompt);

**while**(getchar()!='\n');

}

**return**;

}

**void** Transpose1(**int** \_matrix[][N], **int** n) {

**int** i=0, j=0;

**for** (i=0 ; i<n ; i++) {

**for** (j=0 ; j<n ; j++) printf ("%d ", \_matrix[j][i]);

putchar('\n');

}

}

**void** Transpose2(**int** (\*\_matrix)[N], **int** n) {

**int** i=0, j=0;

**for** (i=0 ; i<n ; i++) {

**for** (j=0 ; j<n ; j++) printf ("%d ", \*(\*(\_matrix+j)+i));

putchar('\n');

}

}

**void** Transpose3(**int** \*\_matrix, **int** n) {

**int** i=0, j=0;

**for** (i=0 ; i<n\*n ; i++) {

printf ("%d ", \*(\_matrix+i));

**if** (i+1%n==0) putchar('\n');

}

}

**int** main() {

**int** matrix[N][N];

**int** i=0, j=0, n=0;

inputNumber(&n, "input n > ");

**for** (i=0 ; i<n ; i++) **for** (j=0 ; j<n ; j++)

scanf("%d", &matrix[i][j]);

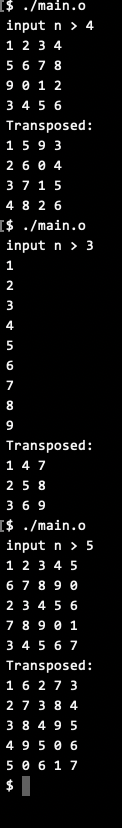
printf("Transposed:\n");

Transpose2(matrix, n);

**return** 0;

}

// Results:



11.5

// Source Codes:

#include <stdio.h>

#define N 12

#define M 12

**void** inputNumber(**int** \*p, **const** **char**\* prompt) {

printf(prompt);

**while**(!scanf("%d", p)) {

printf(prompt);

**while**(getchar()!='\n');

}

**return**;

}

**void** outputArray(**int** \_output[][N], **int** m, **int** n) {

**int** i=0, j=0;

**for** (i=0 ; i<m ; i++) {

**for** (j = 0; j < n; j++) printf("%d ", \_output[i][j]);

putchar('\n');

}

}

**void** Transpose1(**int** \_matrix[][N], **int** \_transposed[][M], **int** m, **int** n) {

**int** i=0, j=0;

**for** (i=0 ; i<n ; i++) **for** (j=0 ; j<m ; j++) \_transposed[i][j] = \_matrix[j][i];

**return**;

}

**void** Transpose2(**int** (\*\_matrix)[N], **int** (\*\_transposed)[M], **int** m, **int** n) {

**int** i=0, j=0;

**for** (i=0 ; i<n ; i++) **for** (j=0 ; j<m ; j++) \*(\*(\_transposed+i)+j) = \*(\*(\_matrix+j)+i);

**return**;

}

**void** Transpose3(**int** \*\_matrix, **int** \*\_transpose, **int** m, **int** n) {

**int** i=0, j=0;

**for** (i=0 ; i<n ; i++) {

**for** (j=0 ; j<m ; j++) \*(\_transpose+(i\*M)+j) = \*(\_matrix+(j\*N)+i);

**if** (i+1%n==0) putchar('\n');

}

**return**;

}

**int** main() {

**int** matrix[M][N];

**int** transposed[N][M];

**int** i=0, j=0, m=0, n=0;

inputNumber(&m, "input m > ");

inputNumber(&n, "input n > ");

**for** (i=0 ; i<m ; i++) **for** (j=0 ; j<n ; j++) scanf("%d", &matrix[i][j]);

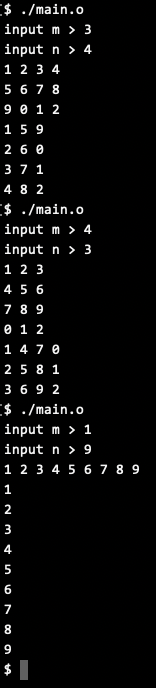
Transpose3(matrix, transposed, m, n);

outputArray(transposed, n, m);

**return** 0;

}

// Results:



11.6

// Source Codes:

#include <stdio.h>

#include <limits.h>

#define M 12

**void** inputNumber(**int** \*p, **const** **char**\* prompt) {

printf(prompt);

**while**(!scanf("%d", p)) {

printf(prompt);

**while**(getchar()!='\n');

}

**return**;

}

**void** inputArray(**int** \*\_input, **int** m, **int** n) {

**int** i=0, j=0;

**for** (i=0 ; i<m ; i++) **for** (j = 0; j < n; j++) inputNumber(\_input+M\*i+j, "");

**return**;

}

**int** findMax(**int** \*\_array, **int** m, **int** n, **int** \*pRow, **int** \*pCol) {

**int** max = INT\_MIN, i=0, j=0;

**for** (i=0 ; i<m ; i++) **for** (j = 0; j < n; j++) **if** (\*(\_array+M\*i+j)>max) {

max = \*(\_array+M\*i+j);

\*pCol = j;

\*pRow = i;

}

**return** max;

}

**int** main() {

**int** col=0, row=0,

m=0, n=0,

array[M][M]={0};

inputNumber(&m, "input m > ");

inputNumber(&n, "input n > ");

inputArray(array, m, n);

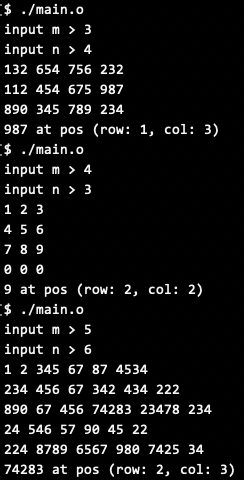
printf ("%d at pos ", findMax(array, m, n, &row, &col));

printf ("(row: %d, col: %d)\n", row, col);

**return** 0;

}

// Results:



11.6

// Source Codes:

#include <stdio.h>

#include <stdlib.h>

#include <limits.h>

#include <string.h>

#include <stdbool.h>

**void** inputNumber(**int** \*p, **const** **char**\* prompt) {

printf(prompt);

**while**(!scanf("%d", p)) {

printf(prompt);

**while**(getchar()!='\n');

}

**return**;

}

**void** inputScore(**int** classNum, **int** stuPerClass, **int** (\*Score)[stuPerClass]) {

**int** i=0, j=0;

**for** (i=0 ; i<classNum ; i++) **for** (j=0 ; j<stuPerClass ; j++) {

printf ("input score of Student NO. %d in Class %d", j+1, i+1);

inputNumber(&Score[i][j], " > ");

}

**return**;

}

**int** Findmax(**int** classNum, **int** stuPerClass, **int** (\*Score)[stuPerClass], **char** (\*Parallels)[stuPerClass]) {

**int** i=0, j=0;

**int** max = INT\_MIN;

**for** (i=0 ; i<classNum ; i++) **for** (j=0 ; j<stuPerClass ; j++) {

**if** (Score[i][j]>max) {

memset (Parallels, -1, classNum\*stuPerClass);

Parallels[i][j] = 1;

max = Score[i][j];

}

**else** **if** (Score[i][j]==max) Parallels[i][j] = 1;

}

**return** max;

}

**int** main() {

**int** classNum=0, stuPerClass=0;

**int** rowPos=-1, colPos=-1;

inputNumber(&classNum, "input Class Number(m) > ");

inputNumber(&stuPerClass, "input student quantity per class(n) > ");

**int** (\*Score)[stuPerClass] = malloc(classNum\*stuPerClass\***sizeof**(**int**));

**char** (\*Parallels)[stuPerClass] = malloc(classNum\*stuPerClass\* **sizeof**(**char**));

inputScore(classNum, stuPerClass, Score);

printf ("\nThe highest mark is %d ", Findmax(classNum, stuPerClass, Score, Parallels));

**int** i=0, j=0;

**bool** firstflag=**true**;

**for** (i=0 ; i<classNum ; i++) **for** (j=0 ; j<stuPerClass ; j++) {

**if** (Parallels[i][j]==1) {

**if** (firstflag) {

printf ("won by Student NO. %d in Class %d (1 as first number)\n", j+1, i+1);

firstflag = **false**;

}

**else** printf ("and his fellow student : Student NO. %d in Class %d\n", j+1, i+1);

}

}

free (Score);

free (Parallels);

**return** 0;

}

// Results:

