

1 OPIS BIBLIOTEKI

Biblioteka «matrix» ma cel przedstawić użytkownikowi funkcji do posługiwania się na macierzach .

Funkcji tej biblioteki faktycznie przejmują znaczenia wejściowych macierzy przez wartość. To pozwala bezpiecznie używać jako wejściowy i wyjściowy parametr tą samą macierz bez wycieków pamięci.

Do tworzenia macierzy trzeba korzystać z funkcji **make_matrix**.

Każdą macierz po wykorzystaniu trzeba zniszczyć za pomocą funkcji **delete_matrix**.

2 OPIS DANYCH

Typ danych	Pola	Opis typu
matrix	Struktura danych, realizująca abstrakcję macierzy.	
	double **array	Macierz dwuwymiarowa typu double . Dostęp do elementów odbywa się jak do zwyczajnej dwuwymiarowej tablicy: array[i][j].
	int n_rows	Liczba wierszy macierzy. Wartość zostaje nadawana przez funkcji biblioteki. Ręczne nadawanie temu polowi wartości spowoduje nieokreślone zachowanie się programu.
	int n_columns	Liczba kolumn macierzy. Wartość zostaje nadawana przez funkcji biblioteki. Ręczne nadawanie temu polowi wartości spowoduje nieokreślone zachowanie się programu.
	int n_permutations	Zawiera liczbę permutacji, które zostały zrobione nad macierzą.
*init_user	Wskaźnik na funkcję o prototypie double foo (int , int). Ta funkcja wykorzystuje się przy inicjalizacji macierzy przez funkcję init_matrix_by_function . Każdemu elementowi macierzy array[i][j] zostaje przepisana wartość foo(int , int), gdzie jako parametry zostają podstawione indeksy bieżącego elementu.	

3 OPIS FUNKCJI

Prototyp funkcji	Opis funkcji
double determinant (IN const matrix*)	Zwraca wartość wyznacznika macierzy, podanej jako parametr. W przypadku błędu zwraca 0, ponieważ zerowy wyznacznik oznacza, że macierz nie jest kwadratowa.
int column_mult_on_const (double factor , int i_column , MODIFIED matrix*)	Mnoży kolumnę <i>i_column</i> macierzy MODIFIED przez wartość <i>factor</i> . W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int columns_sub (double factor , int i_subtracted_column , int i_subtracting_column , MODIFIED matrix*)	Odejmuje od kolumny <i>i_subtracted_column</i> macierzy MODIFIED kolumnę <i>i_subtracting_column</i> , mnożoną przez <i>factor</i> . W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int columns_swap (int i_col_1 , int i_col_2 , MODIFIED matrix*)	Zamienia miejscami kolumny <i>i_col_1</i> i <i>i_col_2</i> macierzy MODIFIED. W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int copy_column_to_other_matrix (int i_source , int i_receiver , IN const matrix *in_matrix , OUT matrix *out_matrix)	Kopiuje kolumnę <i>i_source</i> macierzy <i>in_matrix</i> do kolumny <i>i_receiver</i> macierzy <i>out_matrix</i> . W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int copy_matrix (IN const matrix* , OUT matrix*)	Kopiuje macierz IN do macierzy OUT, zwracając 1 w przypadku sukcesu i 0 w przeciwnym przypadku.
int copy_row_to_other_matrix (int i_source , int i_receiver , IN const matrix *in_matrix , OUT matrix *out_matrix)	Kopiuje wiersz <i>i_source</i> macierzy <i>in_matrix</i> do wierszu <i>i_receiver</i> macierzy <i>out_matrix</i> . W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int delete_matrix (matrix*)	Wyzwala pamięć dynamiczną, zajęta macierzą, podaną jako parametr. W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int init_matrix_as_unit (MODIFIED matrix*)	Inicjalizuje macierz MODIFIED jako jedynkową, to znaczy 1 na głównej przekątnej i 0 na pozostałych miejscach. W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int init_matrix_by_function (MODIFIED matrix* , init_user)	Inicjalizuje każdy element macierzy MODIFIED, funkcją init_user . W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int init_matrix_by_random (MODIFIED matrix* , int down , int up)	Inicjalizuje macierz MODIFIED wartościami losowymi typu int w zakresie od <i>down</i> do <i>up</i> . W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.

int init_matrix (MODIFIED <i>matrix</i> *)	Inicjalizuje macierz, podaną w parametrze wartościami, odczytywanymi ze standardowego wejścia. W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int inverse_matrix (IN <i>const matrix</i> * <i>in_matrix</i> , OUT <i>matrix</i> * <i>out_matrix</i>)	Zapisuje do macierzy <i>out_matrix</i> macierz odwrotną do macierzy <i>in_matrix</i> . W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int multiplex_matrices (IN <i>const matrix</i> * <i>in_matrix_1</i> , IN <i>const matrix</i> * <i>in_matrix_2</i> , OUT <i>matrix</i> *)	Zapisuje do macierzy OUT iloczyn macierzy <i>in_matrix_1</i> przez macierz <i>in_matrix_2</i> . W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int rank (IN <i>const matrix</i> *)	Zwraca rang podanej macierzy.
int row_mult_on_const (<i>double factor</i> , <i>int i_row</i> , MODIFIED <i>matrix</i> *)	Mnoży wiersz <i>i_row</i> macierzy MODIFIED przez wartość <i>factor</i> . W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int rows_sub (<i>double factor</i> , <i>int i_subtracted_row</i> , <i>int i_subtracting_row</i> , MODIFIED <i>matrix</i> *)	Odejmuje od wiersza <i>i_subtracted_row</i> macierzy MODIFIED wiersz <i>i_subtracting_row</i> , mnożony przez <i>factor</i> . W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int rows_swap (<i>int i_row_1</i> , <i>int i_row_2</i> , MODIFIED <i>matrix</i> *)	Zamienia miejscami wierszy <i>i_row_1</i> i <i>i_row_2</i> macierzy MODIFIED. W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int show_matrix (IN <i>const matrix</i> *)	Wypisuje macierz IN w konsoli. W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int transpose (IN <i>const matrix</i> *, OUT <i>matrix</i> *)	Zapisuje do macierzy OUT transponowaną macierz IN. W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
int triangle_form (IN <i>const matrix</i> *, OUT <i>matrix</i> *)	Zapisuje do macierzy OUT macierz IN w postaci schodkowej. W przypadku sukcesu zwraca 1, w przeciwnym przypadku 0.
matrix make_matrix (<i>int n_rows</i> , <i>int n_columns</i>)	Funkcja tworzy nową macierz o <i>n_rows</i> wierszach i <i>n_columns</i> kolumnach.