Листинг 1

#ifndef QUEUE\_H\_INCLUDED

#define QUEUE\_H\_INCLUDED

typedef struct NODE {

int val;

struct NODE \* prev;

struct NODE \* next;

} node\_t;

/\* создание элемента очереди \*/

node\_t \* new\_node(int val){

node\_t \* n;

if ( ( n = (node\_t\*)malloc(sizeof(node\_t)) ) == NULL )

return NULL;

n->val = val;

n->prev = NULL;

n->next = NULL;

return n;

}

/\* удаляет элемент, возвращает указатель на следующий \*/

node\_t \* del\_node (node\_t \* n){

node\_t \* ret = n->next;

if ( n->prev )

n->prev->next = n->next;

if ( n->next )

n->next->prev = n->prev;

free(n);

return ret;

}

/\* очищает список \*/

void clear\_list(node\_t \* n){

while ( n = del\_node(n) )

;

}

/\* удаляет элементы, соответствующие условию, возвращает указатель на первый элемент обработанного списка \*/

node\_t \* remove\_if(node\_t \* n, int (\*test\_func)(const node\_t \* nodeToTest)){

node\_t \* first = n;

while ( first && test\_func(first) )

first = del\_node(first);

if ( ! first )

return NULL;

for ( n = first->next; n != NULL; ){

if ( test\_func(n) )

n = del\_node(n);

else

n = n->next;

}

return first;

}

/\* возращает элемент при его отрицательности \*/

int is\_negative(const node\_t \* n){

return ( n->val < 0 );

}

/\* создаёт список \*/

node\_t \* create\_list(void){

node\_t \* first, \* last, \* current;

static char buf[32] = { 0 };

first = last = NULL;

printf("Enter some numbers (empty string - finish):\n");

while ( fgets(buf, 32, stdin) ) {

if ( \*buf == '\n' )

break;

if ( ( current = new\_node(atoi(buf)) ) == NULL ){

if ( first )

clear\_list(first);

return NULL;

}

if ( ! first ){

first = current;

last = first;

}

else {

current->prev = last;

last->next = current;

last = current;

}

}

return first;

}

/\* Выводит список \*/

void print\_list(const node\_t \* n){

while ( n ){

printf("%d\n", n->val);

n = n->next;

}

}

#endif // QUEUE\_H\_INCLUDED