Linked Lists

This reads numbers from the standard input, and sorts them in increasing numerical order. It then prints the sorted numbers.

/*
* LINKED LIST SORTER
*
* This program reads in numbers and sorts them in increasing numerical order
* The data structure used is a linked list; each element looks like this:
* +--------------+
* | data field  | <--- holds the integer that you read in
* +--------------+
* | next field  | <--- holds pointer to next element in list
* +--------------+       (NULL if nothing follows it)
* The pointer variable "head" contains a pointer to the first element in
* the linked list (NULL if there are no elements in the linked list)
*/
#include <stdio.h>
#include <stdlib.h>

/*
* structure for the list
*/
struct num {
    int data;        /* data field (the number to be sorted) */
    struct num *next; /* points to next element in linked list */
    /* (NULL pointer if no next element) */
};

/*
* pointer to the first element (the head) of the list
* NULL if there's nothing in the list
*/
struct num *head = NULL;

/*
* create a new node
* and initialize the two fields
*/
struct num *createnode(int n)
{
    struct num *p;    /* pointer to new space */

    /* create the element, reporting errors */
    if ((p = malloc(sizeof(struct num))) == NULL)
        return(NULL);

    /* initialize the element */
    p->data = n;
    p->next = NULL;

    /* return a pointer to the new entity */
struct num *insert(struct num *new)
{
    struct num *prev, *temp; /* pointers used to insert new element */

    /* empty list: put head at the front */
    if (head == NULL)
        return(new);

    /* it goes before the first element */
    if (head->data > new->data){
        new->next = head;
        return(new);
    }

    /* now walk the list
     * from here on in, prev->next == temp
     * we'll insert after prev and before temp
     */
    prev = head;
    temp = head->next;
    while(temp != NULL && temp->data < new->data){
        /* advance prev and temp */
        prev = temp;
        temp = temp->next;
    }

    /* here's the insertion
     * make prev->next the new element
     * and new->next the one temp points to
     */
    new->next = temp;
    prev->next = new;

    /* return the pointer to the head of the list */
    return(head);
}

int main(void)
{
    int i; /* number of numbers read by scanf */
    int n; /* what scanf read */
struct num *p; /* pointer to element for linked list */

/*
* loop through the input
*/
while((i = scanf("%d", &n)) != EOF) {

/* error check; was a number read? */
if (i == 0) {
    /* no; give error message and print rest of line */
    fprintf(stderr, "illegal number: ");
    while((i = getchar()) != EOF && i != '\n')
        fputc(i, stderr);
    fputc('
', stderr);
    continue;
}

/* create a new node, and print error message if failure */
if ((p = createnode(n)) == NULL) {
    fprintf(stderr, "no more memory on input %d\n", n);
    return(EXIT_FAILURE);
}

/* insert new element into linked list */
head = insert(p);

/* skip to next line, for cleaner output */
putchar('\n');

/* print the list
* start at the head, print the data field of each element
* and go on to the next
*/
for(p = head; p != NULL; p = p->next)
    printf("%d\n", p->data);

/* bye-bye */
return(EXIT_SUCCESS);}