t_redir->type = (from #define)

>	=	OUTFILE
>>	=	APPEND
<	=	INFILE
<<	=	HEREDOC

t_redir->duplication = (from #define)

>	=	STDOUT_FILENO
>>	=	STDOUT_FILENO
<	=	STDIN_FILENO
<<	=	STDIN_FILENO
cmd |	=	STDOUT_FILENO
| cmd	=	STDIN_FILENO

Duplication refers to the duplication of the file descriptors, to see if we need to do duplication the fd from the standard in (STDIN_FILENO), or if we need to duplicate the file number of the standard output (STDOUT_FILENO).
In the case of a pipe AFTER a command (cmd | …) the *output* of the command needs to be redirected to the pipe, so the redirection->duplication of the pipe redirection node is = STDOUT_FILENO.

ONE COMMAND (NO REDIRECTION)

ls

```
t_exec *exec_struct;
t_cmd  *cmd_node;

malloc_new_command_node(cmd_node); //example of malloc function for t_cmd
cmd_node->array = { “ls”, NULL};
cmd_node->redir = NULL;
cmd_node->next = NULL;

exec->cmd = cmd_node;
```

ONE COMMAND (ONE FILE REDIRECTION)

echo -n -nn “Hello world” > file1

```
t_exec *exec_struct;
t_cmd  *cmd_node;
t_redir  *redir_node;

malloc_new_redir_node(redir_node); //example of malloc function for t_redir
redir_node->type = OUTFILE;
redir_node->file_name = “file1”;
redir_node->duplication = STDOUT_FILENO;
redir_node->next = NULL;

malloc_new_command_node(cmd_node);
cmd_node->array = { “echo”, “-n”, “-nn”, “Hello  world”, NULL};
// or this could also work: cmd_node->array = { “echo”, “-n”, “-nn”, “Hello”, “ “,  “world”, NULL};
cmd_node->redir = redir_node;
cmd_node->next = NULL;

exec->cmd = cmd_node;
```

1 COMMAND (3 FILE REDIRECTIONS)

file1 < ls << heredoc >> file2

redirection nodes start from left to right, < is the head of the redir list, and >> would be the tail

```
t_exec *exec_struct;
t_cmd  *cmd_node;
t_redir  *redir_node1;
t_redir  *redir_node2;
t_redir  *redir_node3;

malloc_new_redir_node(redir_node_1);
redir_node1->type = INFILE;
redir_node1->file_name = “file1”;
redir_node1->duplication = STDIN_FILENO;

malloc_new_redir_node(redir_node_2);
redir_node2->type = HEREDOC;
redir_node2->heredoc_buff = //string containing the text from the heredoc;
redir_node2->duplication = STDIN_FILENO;
redir_node1->next = redir_node2;

malloc_new_redir_node(redir_node_3);
redir_node3->type =APPEND;
redir_node3->file_name = “file2”;
redir_node3->duplication = STDOUT_FILENO;
redir_node2->next = redir_node3;
redir_node3->next = NULL;

malloc_new_command_node(cmd_node);
cmd_node->array = { “ls”, NULL};
cmd_node->redir = redir_node1;
cmd_node->next = NULL;

exec->cmd = cmd_node;
```

2 COMMANDS (1 PIPE REDIRECTIONS)

ls -l | wc

redirection nodes start from left to right, < is the head of the redir list, and >> would be the tail

```
t_exec *exec_struct;
t_cmd  *cmd_node1;
t_redir  *redir_node1;
t_cmd  *cmd_node2;
t_redir  *redir_node2;

//        ls -l      |
malloc_new_redir_node(redir_node_1);
redir_node1->type = PIPE;
redir_node1->duplication = STDOUT_FILENO;
redir_node1->next = NULL;
//duplication is the standard output because the output of ls -l is “sent” to the pipe;

malloc_new_command_node(cmd_node1);
cmd_node1->array = { “ls”, “-l”, NULL};
cmd_node1->redir = redir_node1;

//        |          wc
malloc_new_redir_node(redir_node_2);
redir_node2->type = PIPE;
redir_node2->duplication = STDIN_FILENO;
redir_node2->next =NULL;
//duplication is the standard input because the input of wc is “received” from the pipe;

malloc_new_command_node(cmd_node2);
cmd_node2->array = { “wc”, NULL};
cmd_node2->redir = redir_node2;
cmd_node2->next = NULL;
cmd_node1->next = cmd_node2;

exec->cmd = cmd_node1;
```

3 COMMANDS (2 PIPES REDIRECTIONS)

ls | wc | grep “0”

```
//      ls      |
malloc_new_redir_node(redir_node_1);
redir_node1->type = PIPE;
redir_node1->duplication = STDOUT_FILENO;
redir_node1->next = NULL;

malloc_new_command_node(cmd_node1);
cmd_node1->array = { “ls”, NULL};
cmd_node1->redir = redir_node1;

//      |       wc      |
malloc_new_redir_node(redir_node_1);
redir_node1->type = PIPE;
redir_node1->duplication = STDIN_FILENO;

malloc_new_redir_node(redir_node_2);
redir_node2->type = PIPE;
redir_node2->duplication = STDOUT_FILENO;
redir_node2->next = NULL;
redir_node1->next = redir_node2;

malloc_new_command_node(cmd_node2);
cmd_node2->array = { “wc”, NULL};
cmd_node2->redir = redir_node1;
cmd_node1->next = cmd_node2;

//      |       grep “0”
malloc_new_redir_node(redir_node_1);
redir_node1->type = PIPE;
redir_node1->duplication = STDIN_FILENO;
redir_node1->next = NULL;

malloc_new_command_node(cmd_node3);
cmd_node3->array = { “grep”, “0”, NULL};
cmd_node3->redir = redir_node1;
cmd_node3->next = NULL;
cmd_node2->next = cmd_node3;

exec->cmd = cmd_node1;
```

3 COMMANDS (MANY REDIRECTIONS)

File1 < ls > file2 | heredoc << wc >> file3 | < file4 grep 0

```
//      File1 < ls > file2 |
malloc_new_redir_node(redir_node_1);
redir_node1->type = INFILE;
redir_node1->file_name = “File1”;
redir_node1->duplication = STDIN_FILENO;

malloc_new_redir_node(redir_node_2);
redir_node2->type =OUTFILE;
redir_node2->file_name = “file2”;
redir_node2->duplication = STDOUT_FILENO;
redir_node1->next =redir_node2;

malloc_new_redir_node(redir_node_3);
redir_node3->type = PIPE;
redir_node3->duplication = STDOUT_FILENO;
redir_node3->next = NULL;
redir_node2->next = redir_node3;

malloc_new_command_node(cmd_node1);
cmd_node1->array = { “ls”, NULL};
cmd_node1->redir = redir_node1;

//      |       heredoc << wc >> file3       |

malloc_new_redir_node(redir_node_1);
redir_node1->type = PIPE;
redir_node1->duplication = STDIN_FILENO;

malloc_new_redir_node(redir_node_2);
redir_node2->type = HEREDOC;
redir_node2->heredoc_buffer =  //string containing the text from the heredoc;
redir_node2->duplication = STDIN_FILENO;
redir_node1->next =redir_node2;

malloc_new_redir_node(redir_node_3);
redir_node3->type = APPEND;
redir_node3->file_name = “file3”;
redir_node3->duplication = STDOUT_FILENO;
redir_node2->next =redir_node3;

malloc_new_redir_node(redir_node_4);
```

```
redir_node4->type = PIPE;
redir_node4->duplication = STDOUT_FILENO;
redir_node4->next = NULL;
redir_node3->next = redir_node4;

malloc_new_command_node(cmd_node2);
cmd_node2->array = { “wc”, NULL};
cmd_node2->redir = redir_node1;
cmd_node1->next = cmd_node2;

//          | < file4 grep 0
malloc_new_redir_node(redir_node_1);
redir_node1->type = PIPE;
redir_node1->duplication = STDIN_FILENO;

malloc_new_redir_node(redir_node_2);
redir_node2->type = INFILE;
redir_node2->file_name = “file4”;
redir_node2->duplication = STDIN_FILENO;
redir_node2->next = NULL;
redir_node1->next = redir_node2;

malloc_new_command_node(cmd_node3);
cmd_node3->array = { “grep”, “0”, NULL};
cmd_node3->redir = redir_node1;
cmd_node3->next = NULL;
cmd_node2->next = cmd_node3;

exec->cmd = cmd_node1;
```