Assignment UP-01

In process.c,

Data Structure:

```
int argc;
char *argv[]];
int bytes_written;
char *addr[];
```

Functions added:

```
Void test_stack (int *t)
// It is used to print contents in the stack
```

Functions modified:

```
Tid_t process_execute (const char *file_name)
//  For synchronizations between threads.

Bool load (const char *cmd_line_input, void (**eip) (void), void **esp)
// called test_stack from it.

static bool setup_stack (void **esp, char *file_name, char *args)
// Arranged the stack as given in the pintDOC.
```

Algorithm:

1. Firstly we split the total argument into filename and different arguments with the use of strtok_r() function.
2. Then we added arguments in the stack, saving their corresponding addresses in the array.
3. We added word align in the stack and then added addresses of the arguments in the stack.
4. FInally we added address of argv and argc into the stack followed by return address.

This allowed us to make stack properly, but we had difficulty in arranging of the contents of the result. When the thread was created in process_execute function, it was expected from us that the new thread created should run by calling thread_yield as otherwise execution order of answer won't be correct. To tackle this problem, we made the new thread priority high as our thread implementation from previous assignments included thread priority as a criteria to which thread will run next.