```bash
#!/bin/bash

################################################################################
# Description for the intranet check (one line, support Markdown syntax)
# Remove all environment variables and execute `env`

################################################################################
# The variable 'compare_with_sh' IS OPTIONNAL
#
# Uncomment the following line if you don't want the output of the shell
# to be compared against the output of /bin/sh
#
# It can be useful when you want to check a builtin command that sh doesn't
# implement
# compare_with_sh=0

################################################################################
# The variable 'shell_input' HAS TO BE DEFINED
#
# The content of this variable will be piped to the student's shell and to sh
# as follows: "echo $shell_input | ./hsh"
#
# It can be empty and multiline
shell_input="env"

################################################################################
# The variable 'shell_params' IS OPTIONNAL
#
# The content of this variable will be passed to as the paramaters array to the
# shell as follows: "./hsh $shell_params"
#
# It can be empty
# shell_params=""

################################################################################
# The function 'check_setup' will be called BEFORE the execution of the shell
# It allows you to set custom VARIABLES, prepare files, etc
# If you want to set variables for the shell to use, be sure to export them,
# since the shell will be launched in a subprocess
#
# Return value: Discarded
function check_setup()
{
        current_env=$(/usr/bin/env)
        for i in `/usr/bin/env | /usr/bin/cut -d'=' -f1`
        do
                unset $i
        done

        # Important: Disable valgrind when running without an environment
        let valgrind_error=0
        let valgrind_leak=0

        return 0
}

################################################################################
# The function 'sh_setup' will be called AFTER the execution of the students
# shell, and BEFORE the execution of the real shell (sh)
# It allows you to set custom VARIABLES, prepare files, etc
# If you want to set variables for the shell to use, be sure to export them,
# since the shell will be launched in a subprocess
#
# Return value: Discarded
function sh_setup()
{
        return 0
}

################################################################################
# The function `check_callback` will be called AFTER the execution of the shell
# It allows you to clear VARIABLES, cleanup files, ...
#
# It is also possible to perform additionnal checks.
# Here is a list of available variables:
# STATUS -> Path to the file containing the exit status of the shell
# OUTPUTFILE -> Path to the file containing the stdout of the shell
# ERROR_OUTPUTFILE -> Path to the file containing the stderr of the shell
# EXPECTED_STATUS -> Path to the file containing the exit status of sh
# EXPECTED_OUTPUTFILE -> Path to the file containing the stdout of sh
# EXPECTED_ERROR_OUTPUTFILE -> Path to the file continaing the stderr of sh
#
# Parameters:
#     $1 -> Status of the comparison with sh
#             0 -> The output is the same as sh
#             1 -> The output differs from sh
#
# Return value:
#     0  -> Check succeed
#     1  -> Check fails
function check_callback()
{
        let status=0

        # Remove environment variables and set by valgrind from student output
        content=`$CAT "$OUTPUTFILE"`
        content=`$ECHO "$content" | $GREP -v -e "^GLIBCPP_FORCE_NEW="`
        content=`$ECHO "$content" | $GREP -v -e "^GLIBCXX_FORCE_NEW="`
        content=`$ECHO "$content" | $GREP -v -e "^LD_PRELOAD="`
        content=`$ECHO "$content" | $GREP -v -e "^LD_LIBRARY_PATH="`
        content=`$ECHO "$content" | $GREP -v -e "^_="`
        content=`$ECHO "$content" | $GREP -v -e "^PWD="`
        $ECHO "$content" > $OUTPUTFILE

        # Remove "_" environment variable from expected output
        content=`$CAT "$EXPECTED_OUTPUTFILE"`
        content=`$ECHO "$content" | $GREP -v -e "^_="`
        content=`$ECHO "$content" | $GREP -v -e "^PWD="`
        $ECHO "$content" > $EXPECTED_OUTPUTFILE

        $ECHO -n "" > $EXPECTED_ERROR_OUTPUTFILE
        $ECHO -n "0" > $EXPECTED_STATUS

        check_diff

        return $status
}
```