SCALE FOR PROJECT MINISHELL (/PROJECTS/42CURSUS-MINISHELL)

You should evaluate 2 students in this team

★

Git repository

```
git@vogsphere.42urduliz.com:vogsphere/intra-uuid-089f8429-b7b4-4f8c-
```

Introduction

Por favor respeta las siguientes normas:

- Sé educado, coordial, respetuoso y constructivo durante el proceso de evaluación. El bienestar de la comunidad depende de ello.

- Identifica con la persona o grupo evaluado, las posibles disfuncionalidades del trabajo. Tómate el tiempo de discutir y debatir los problemas que puedes haber identificado.

- Debes considerar la posibilidad de diferir en el entendimiento de las instrucciones del proyecto y el alcance de la funcionalidad. Mantén siempre una mente abierta y evalúa de forma honesta. La pedagogía sólo es válida si las evaluaciones se toman con seriedad.

Guidelines

- Evalúa sólo el trabajo entregado en el repositorio Git del estudiante o grupo.

- Comprueba dos veces que el repositorio Git pertenece al estudiante o grupo evaluado. Asegúrate de que el trabajo entregado es el solicitado por el proyecto y que "git clone" es utilizado en una carpeta vacía.

- Comprueba cuidadosamente que no existan alias maliciosos usados para engañarte y hacerte evaluar trabajo subido a un repositorio no oficial.

- Para evitar sorpresas, comprueba cuidadosamente que tanto el estudiante evaluado como el evaluador han revisado los posibles scripts utilizados durante la entrega para facilitar la evaluación.

- Si el evaluador no ha completado este proyecto todavía, es obligatorio para el estudiante leer el subject entero antes de empezar la defensa.

- Utiliza las flags disponibles en esta evaluación para señalar un repositorio vacío, un programa disfuncional, un fallo de norma, trampas, etc. En estos casos, la evaluación termina y la nota final es 0 (o -42 en caso de trampa). Sin embargo, a excepción de trampa, se recomienda seguir discutiendo el trabajo (incluso si no está terminado) para identificar posibles fallos y evitar repetirlos en el futuro.

- Recuerda que durante la defensa, ningún segfault, ni otros comportamientos prematuros, descontrolados o cierres del programa se toleran. En caso contrario, la nota final es 0. Utiliza la flag apropiada.
Nunca debes editar ningún archivo salvo el de configuración si existe.
Si quieres editar un archivo, tómate el tiempo de explicar las razones al estudiante evaluado y asegúrate de que ambos estáis de acuerdo.

- Debes también verificar la ausencia de leaks. Toda la memoria localizada en el heap debe liberarse previamente al final de la ejecución.
Tienes permitido usar distintas herramientas disponibles en el ordenador, tales como leaks, valgrind o e_fence. En caso de leaks, utiliza la flag apropiada.

Attachments

subject.pdf (https://cdn.intra.42.fr/pdf/pdf/31394/es.subject.pdf)

Parte obligatoria

Compila

- Utiliza `make -n` para verificar que la compilación utilice -Wall -Werror -Wextra, si no utiliza la flag de compilación inválida.

- `minishell` compila sin errores, si no utiliza la flag apropiada.

- El Makefile no debe hacer relink.

Yes No

Comados simples y la variable global

- Ejecuta un comando sencillo con una ruta absoluta de la forma /bin/ls, o cualquier otro comando sin argumentos.

- ¿Cuántas variables globales utiliza? ¿Por qué? Debe dar un ejemplo concreto de por qué le parece necesario o lógico.

- Prueba un comando vacío.

- Prueba solo tabuladores o espacios.

- Si el programa termina, utiliza la flag de crash.

- Si algo no funciona, utiliza la flag de trabajo incompleto.

Yes No

Argumentos e historial

- Ejecuta un comando simple con una ruta absoluta como /bin/ls, u otro comando con argumentos pero sin comillas simples y comillas dobles.

- Hazlo varias veces con distintos comandos y argumentos.

- Si el programa termina, utiliza la flag de crash.

- Si algo no funciona, utiliza la flag de trabajo incompleto.

Yes No

echo

- Ejecuta el comando echo con o sin argumentos, y con o sin -n.

- Hazlo múltiples veces con distintos argumentos.

- Si el programa termina, utiliza la flag de crash.

- Si algo no funciona, utiliza la flag de trabajo incompleto.

Yes No

exit

- Ejecuta el comando exit con o sin argumentos.

- Hazlo varias veces con distintos argumentos.

- No olvides lanzar de nuevo minishell.

- Si el programa termina, utiliza la flag de crash.

- Si algo no funciona, utiliza la flag de trabajo incompleto.

Yes No

Valor de retorno de un proceso

- Ejecuta una prueba simple con una ruta absoluta del tipo /bin/ls, o algún otro comando con argumentos pero sin comillas simples o comillas dobles. Después ejecuta echo $?

- Comprueba el valor devuelto. Puedes hacer lo mismo en bash y comparar ambos resultados.

- Hazlo varias veces, con distintos comandos y argumentos. Ejecuta algunos comandos que fallen como '/bin/ls archivo_que_no_existe'.

- Algo como 'expr $? + $?'.

- Si el programa termina, utiliza la flag de crash.

- Si algo no funciona, utiliza la flag de trabajo incompleto.

Yes No

Señales

- Verifica que ctrl-C en una entrada limpia muestra una nueva línea con una entrada limpia.
- Verifica que ctrl-\ en una entrada limpia no hace nada.
- Verifica que ctrl-D en una entrada limpia termina minishell, ejecútalo de nuevo.
- Verifica que ctrl-C en una entrada con texto, muestra una nueva línea con una entrada limpia.
- El buffer debería limpiarse correctamente, pulsa "enter" para verificar que nada de lo que habías escrito se ejecute.
- Verifica que ctrl-D en una entrada con texto no hace nada.
- Verifica que ctrl-\ en una entrada con texto no hace nada.
- Verifica que ctrl-C después de ejecutar un comando bloqueante como cat sin argumentos o grep "algo".
- Verifica que ctrl-\ después de ejecutar un comando bloqueante como cat sin argumentos o grep "algo".
- Verifica que ctrl-D después de ejecutar un comando bloqueante como cat sin argumentos o grep "algo".
- Hazlo varias veces con distintos comandos.
- Si el programa termina, utiliza la flag de crash.
- Si algo no funciona, utiliza la flag de trabajo incompleto.

Yes No

Comillas dobles

- Ejecuta un comando simple con argumentos, esta vez con comillas dobles (deberás incluir espacios).

- Un comando como echo "cat lol.c | cat > lol.c"

- Cualquier cosa, salvo $.

- Si el programa termina, utiliza la flag de crash.

- Si algo no funciona, utiliza la flag de trabajo incompleto.

Yes No

Comillas simples

- Ejecuta comandos con comillas simples como argumento.
- Prueba argumentos vacíos.
- Prueba variables de entorno, espacios en blanco, pipes y redirecciones en las comillas simples.
- echo '$USER' debe imprimir $USER.
- Nada debe interpretarse.

Yes No

env

- Comprueba si env muestra las variables de entorno actuales

Yes No

export

- Exporta variables de entorno, crea nuevas, y reemplaza viejas.
- Comprueba que funcione correctamente con env.

Yes No

unset

- Exporta variables de entorno, crea nuevas, y reemplaza viejas.
- Utiliza unset para eliminar algunas de ellas.
- Verifica el resultado con env.

Yes No

cd

- Utiliza el comando cd para cambiar de directorio de trabajo, y asegúrate de que estás en el directorio correcto con /bin/ls.

- Repite esto varias veces, algunas que funcionen y otras que no.

- Prueba '.' y '..' como argumentos.

Yes No

pwd

- Utiliza el comando pwd.

- Repite esto varias veces en múltiples directorios.

Yes No

Rutas relativas

- Ejecuta comandos, pero esta vez con rutas relativas.

- Repite esto múltiples veces en múltiples directorios con rutas relativas complejas (muchos ..)

Yes No

La variable de entorno PATH

- Ejecuta comandos, pero esta vez sin rutas (ls, wc, awk, etc).

- Elimina $PATH y verifica que ya no funcionen.

- Establece $PATH para varios directorios (directorio1:directorio2) y valida que los directorios se comprueban de izquierda a derecha.

Yes No

Redirecciones

- Ejecuta comandos con redirecciones < y/o >.

- Repite esto varias veces con diferentes comandos y argumentos, cambia de vez en cuando > por >>.

- Comprueba si varias redirecciones del mismo tipo fallan.

- Prueba la redirección << (no necesita actualizar el historial).

Yes No

Pipes

- Ejecuta comandos con pipes como 'cat file | grep bla | more'.

- Haz esto varias veces cambiando comandos y argumenetos.

- Prueba algunos comandos que fallen como 'ls archivoquenoexiste | grep bla | more'.

- Prueba a mezclar pipes y redirecciones.

Yes No

Vuélvete loco, y el historial

- Escribe una línea de comandos, utiliza ctrl-C y pulsa enter. El buffer deberá estar vacío y nada deberá ejecutarse.
- ¿Puedes navegar en el historial con las flechas de arriba y abajo para reintentar algún comando?
- Ejecuta comandos que no deberían funcionar como 'wjkgjrgwg4g43go34o' y verifica que minishell no termina y muestra un error.
- "cat | cat | ls" se comporta de forma "normal".
- Prueba a ejecutar un comando largo con muchísimos argumentos.
- Pásatelo bien con este increíble minishell y disfruta.

Yes No

Variables de entorno

- Ejecuta echo con algunas $ variables como argumentos.
- Verifica que $ se interpreta como una variable de entorno.
- Verifica que las comillas dobles interpolan $.
- Verifica que $USER existe o créalo.
- echo "$USER" deberá imprimir el valor de $USER.

Yes No

Extras

Se tendrá en cuenta tu parte extra si y solo si la parte obligatoria está PERFECTA. Por PERFECTA queremos naturalmente decir que la parte obligatoria está completa, de principio a fin, y la gestión de errores es impecable, incluso en casos retorcidos o de mal uso. Si no has conseguido todos los puntos de la parte obligatoria, la parte bonus será completamente ignorada.

And, Or

- Utiliza &&, || y paréntesis con comandos y verifica que funciona como en bash.

Yes No

WildCard

- Utiliza wildcards en argumentos para el directorio actual.

Yes No

¡Sorpresa! (O no...)

- Establece la variable de entorno USER.

- Prueba echo "'$USER'", esto deberá imprimir 'VALOR_DE_USER'.

- Prueba echo "'$USER'", deberá imprimir "$USER".

Yes No

Ratings

Don't forget to check the flag corresponding to the defense

Ok Outstanding project

Empty work No author file Invalid compilation Norme Cheat Crash Leaks

Forbidden function

Conclusion

Leave a comment on this evaluation

Finish evaluation