1. Instrucciones

La tarea es individual, debe resolverla e investigar por su propia cuenta. Cualquier intento de plagio se procesará de acuerdo al reglamento de la Universidad de Costa Rica.

La fecha de entrega es el día lunes 22 de octubre de 2018 a las 23:55.

Entregables: Debe entregar en el sitio virtual del curso, un único archivo con extensión .s, el archivo debe llamarse <carné>_tarea<número de tarea>_grupo<número de grupo>.s, además el archivo debe contener un pequeño encabezado con sus datos y explicación breve del código implementado.

Es sumamente necesario que el código contenga **comentarios** que explican el porqué de lo realizado.

La función main debe contener **únicamente** llamadas a otras funciones y no grandes bloques de código.

Su código debe ser ejecutable en **QTSpim**.

2. Calculadora Polaca Inversa

En esta tarea usted implementará una calculadora usando la notación polaca inversa (RPN por sus siglas en inglés). Esta calculadora será capaz de realizar las siguientes operaciones:

- suma
- resta
- multiplicación
- división
- elevar al cuadrado
- cálculo de la raíz cuadrada

Una calculadora que utiliza la RPN hace uso de la pila para realizar las operaciones, primero se introducen los operandos y luego la operación. Por ejemplo para hacer la operación 241 + 7, se haría lo siguiente:

241 Se apila 241

7 Se apila 7

+ Se detecta un operando, se desapilan el 7 y el 241 y se suman, se apila el resultado (248)

La siguiente secuencia 2 5 8 + * haría lo siguiente:

2 Se apila el 2

5 Se apila el 5

8 Se apila el 8

+ Se desapilan el 8 y el 5, se suman y se apila el resultado (13)

* Se desapilan el 13 y el 2, se multiplican y se apila el resultado (26)

Otros ejemplos:

- (1+2)×(3+4) sería en RPN 1 2 + 3 4 + × = 21
- 1+2×3+4 sería en RPN 1 2 3 × 4 + = 11
- 1+2×(3+4) sería en RPN 1 2 3 4 + × + = 15
- (1+2)×3+4 sería en RPN 1 2 + 3 × 4 + = 13

Para hacer esta calculadora el simulador siempre deberá estar esperando operandos, estos serán introducidos como un *string*:

```
.data
input: .asciiz " "

.text
la $a0, input
li $v0, 8 #$a0=dirección del buffer donde se almacena el string
syscall
```

El *string* que se recibe se debe procesar de la siguiente forma:

- El inicio está en $a0
- Se debe determinar el tamaño del *string* (número de caracteres=NUMCHAR), el primer carácter está en 0($a0) y el último es 0x0A.
- El último carácter ingresado está en NUMCHAR-2+$a0, NUMCHAR-1+$a0 es igual a 0x0A.
- Se deben procesar los caracteres ingresados (estos están en código ASCII, space:32, 0:48 - 9:57, +:43, -:45, *:42, /:47, C:67, c: 99, null:0, P:80 p:112, S:83, s:115),
    - Si es una operación básica +, -, *, /, s o S (raíz cuadrada), p o P (elevar al cuadrado) se debe ejecutar según la RPN. Note que las operaciones se ejecutan de inmediato, estas no se apilan.

- Si es un espacio debe mostrar el último resultado (último elemento apilado), hay que tener cuidado de no eliminar el elemento de la pila, solo mostrarlo.
- Si es C o c debe borrar toda la pila y poner cero como último elemento apilado.
- Si es un dígito (0-9) debe procesar el *string* para calcular el número. Por ejemplo si se ingresa 978, al procesar el *string* se determina que NUMCHAR es 4, se lee el byte almacenado en NUMCHAR-2+$a0, se determina que corresponde al entero 56 (carácter 8), como es un elemento válido y un dígito, se le resta 48 (carácter 0) lo que da como resultado 8. Se procesa el byte almacenado en NUMCHAR-3+$a0 de la misma forma que el carácter anterior, se determina que corresponde al entero 7, este se multiplica por 10 y se suma al resultado anterior. Se procesa el byte almacenado en NUMCHAR-4+$a0 de la misma forma que el carácter anterior, se determina que corresponde al entero 9, este se multiplica por 100 y se suma al resultado anterior. El resultado 978 se envía a un registro de punto flotante y se apila el resultado.

  Debe tener cuidado ya que al inicio del *string* puede haber un -, indicando que se trata de un número negativo. Puede suponer que los números ingresados son siempre enteros, aunque con signo.

En el caso que se ingrese un carácter no válido debe imprimirse un mensaje de error y detener la ejecución. Por ejemplo:

```
err: .asciiz "\n\nCaracter no válido, acaba de morir un gatito\n"
error:
     li $v0, 4
     la $a0, err
     syscall
     li $v0, 10
     syscall
```

Para realizar esta tarea sea ordenado, tome tiempo antes de programar para realizar una planificación de las funciones que requiere. Utilice la instrucción *jal* para usar estas funciones y *jr* para regresar de estas, así como los registros adecuados, a_x$ para pasar parámetros a las funciones y v_x$ para obtener los resultados de estas, en el caso de las funciones de punto flotante puede usar los registros f_x$.

Entre las funciones que debería tener están:

`getNUMCHAR` devuelve en $v0 el número de caracteres del *string*.

`getNumber` devuelve el número almacenado en el *string* en el registro $v0

Opcional 20 puntos extra

Haga que la calculadora no solo reciba enteros, sino que trabaje también con flotantes. Al detectar un número se deberá localizar la coma decimal (,:44) e ir dividiendo entre 10 al moverse a la derecha y multiplicando por 10 hacia la izquierda.