Arquitectura y Organización de Computadores
Tarea 1

Profesor Mauricio Solar
Ayudantes Joaquín Montes
Sebastián Alvarado
Felipe González
Javier Rojas

1 Reglas Generales

Para esta tarea deberá utilizar el lenguaje de programación Python para escribir un programa capaz de completar los objetivos mencionados en la sección de enunciado. Se recomienda usar Python 3.7 o superior, pero cualquier versión de Python 3 debiera funcionar para desarrollar la tarea.

Además deberá confeccionar un informe detallando el desarrollo de su tarea. El informe debe entregarse en formato PDF y contener las secciones Portada, Resumen, Introducción, Desarrollo, Resultados, Análisis, y Conclusión. Se recomienda utilizar LaTeXpara construir el informe, márgenes de 25 milímetros, y mantener el orden haciendo uso de secciones y subsecciones.

2 Enunciado

La empresa *Ocular*^{TM} lo contrató para desarrollar el futuro de la entretención digital: *Abyss*, una consola de videojuegos de inmersión completa, que captura todos los sentidos del jugador y le permite disfrutar de un mundo digital ultra realista.

Su primera tarea consiste en crear un programa que traduzca los distintos tipos de código que utilizará la consola, desde código en unario (base 1), hasta código en base 64. Se le ha otorgado una estación de trabajo con software de vanguardia, incluyendo el nuevo lenguaje de programación que revolucionó al mundo, Python 3. Utilice los algoritmos que conoce para convertir números entre dos bases cualesquiera.

Ocular^{TM} se preocupa mucho de la seguridad de sus sistemas, por lo que además le solicita que implemente los códigos BCD, Gray, Exceso de 3, Johnson, Paridad, PentaBit y Hamming; todos estos tomando números en binario pero interpretándolos a su manera. En caso de recibir un número decimal (es decir, que contenga dígitos mayores a 1), primero debe convertir el número a binario antes de interpretarlo con el código seleccionado. Las bases numéricas entre 1 y 64 se expresan con su valor, mientras que los códigos siguen la siguiente tabla.

Código	Expresión
BCD	`bcd`
Gray	`gry`
Exceso de 3	`ed3`
Johnson	`jsn`
Paridad	`par`
PentaBit	`pbt`
Hamming	`ham`

3 Entrada y salida de datos

La entrada de datos se hará a través de la entrada estándar (`stdin`) en tríos de datos '`n b t`' ($1 \leq$ `n` $\leq 1000; 1 \leq b, t \leq 64$, o uno de los siguientes códigos: `bcd`, `gry`, `ed3`, `jsn`, `par`, `pbt`, `ham`), donde `n` es el número a convertir, `b` es la base o código en que se encuentra, y `t` es la base o código objetivo a lo que convertir. Debe validar que los datos entregados sean correctos. Nótese que en la entrada, el valor de `n` va a ser siempre menor o igual a 1000 *al pasarlo a decimal*.

La salida de datos se hará a través de la salida estándar (`stdout`) siguiendo el formato '`Base x: r`' o el formato '`Código z: r`' dependiendo de la base o código objetivo, donde `x` es la base objetivo, `z` es el nombre del código, y `r` es el resultado de convertir el número. En caso de encontrar datos inválidos se debe imprimir '`Entrada invalida`' en vez de presentar el resultado.

Las bases entre 1 y 10 usan los dígitos decimales, mientras que las bases entre 11 y 64 usan una partición de izquierda a derecha de los siguientes símbolos. Se incluye un ejemplo para mejor claridad. Todos los códigos de la Tabla 1 expresan sus valores en binario.

`0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ+?`

Ejemplo: Base 21 usaría los símbolos '`0123456789abcdefghijk`'.

Cuando la base de entrada `b` sea un código, el valor `n` entregado debe ser primero convertido a base 2. Para esta conversión, asuma lo siguiente:

- Si `n` sólo incluye los dígitos 0 y 1, y además comienza con el dígito 0, la base de `n` es 2. Para la longitud del valor, el primer 0 se ignora. Así, el valor 011 tendría una longitud de 2 bits.
- Si `n` no comienza con 0, y todos sus dígitos son números, la base de `n` es 10.
- Si `n` contiene algún dígito que no sea un número, la base de `n` es desconocida y no se puede procesar. En este caso, los datos se consideran inválidos, y debe imprimir la respuesta correspondiente.

Para todos los códigos salvo `BCD`, el valor resultante se considerará como un número en base 2. Para `BCD`, el valor resultante se considerará en base 10.

Tenga las siguientes consideraciones para códigos específicos:

- **Johnson.** Al convertir a código Johnson, la longitud del código debe ser la menor potencia de 2 suficiente.
- **Paridad.** El valor de salida debe ser 0 si se detecta un error, 1 si el valor es correcto. Se asume que el valor no tiene errores si la cantidad de 1s es par.
- **PentaBit.** El valor de salida debe ser 0 si se detecta un error, 1 si el valor es correcto. Si la longitud del código no es múltiplo de 5, la entrada es inválida.
- **Hamming.** Si se detecta un error, el valor de salida debe ser el valor corregido. Si el valor no se puede corregir, la salida debe ser 0.

4 Datos de ejemplo

A continuación se presenta una serie de entradas y sus respectivas salidas tras ser procesadas por el programa.

stdin	**stdout**
`1010 2 10`	`Base 10:  10`
`1337 8 2`	`Base 2:  1011011111`
`221 3 bcd`	`Código BCD: 19`
`17 bcd 10`	`Base 10:  11`
`011 gry 2`	`Base 2:  10`

Explicación de los ejemplos
`1010 2 10`: 1010 de binario a decimal. Usamos el algoritmo estándar para convertir el número, obteniendo el número 10 en base 10. Nótese que el valor de entrada no comienza con un 0: como la base de entrada se especifica, no es necesario indicarlo con el cero precedente.

`1337 8 2`: 1337 de octal a binario. Descomponemos el número en octal y convertimos usando el algoritmo estándar, obteniendo el número 1011011111 en base 2. Nótese que el valor de entrada, 1337, cumple con la restricción inicial de que $n \leq 1000$, pues la base de entrada es *octal*. Al transformar a base 10, se ve que el valor entregado está dentro del rango válido.

`221 3 bcd`: 221 de ternario al código BCD. Primero convertimos a binario usando el algoritmo estándar, obteniendo el número 11001 en base 2. Luego extendemos la cantidad de dígitos del resultado a 00011001 para poder usar BCD, y obtenemos cada dígito del resultado en base 10 para obtener 19.

`17 bcd 10`: 17 de código BCD a decimal. Primero debemos convertir el 17 en base 10 a un valor en base 2 para trabajar con BCD, obteniendo el valor 10001. Este valor se separa en grupos de 4 bits para convertir cada uno a su dígito decimal correspondiente, obteniendo así el valor 11. Formalmente, el paso siguiente es transformar el valor entregado por BCD a la base de salida, pero en este caso son iguales y se imprime directamente.

`011 gry 2`: Nótese que el valor de entrada está en base 2, pues comeinza con un 0 y sus dígitos son sólo 0s y 1s. Dado esto, se considera directamente como un código Gray. Convertimos el número directamente, notando que el código 11 equivale al segundo "cambio" en la tabla de códigos de Gray, entonces el resultado es 10. Finalmente, se convierte el valor 10 de base 2 a la base de salida, que en este caso también es 2, por lo que se imprime directamente.

5 Consideraciones

- La tarea debe realizarse individualmente. Ante cualquier sospecha de copia o trabajo colaborativo se informará a las autoridades correspondientes.
- La tarea debe realizarse utilizando un editor de texto plano o código cualquiera. El código debe estar escrito usando el lenguaje de programación Python 3.x. Se recomienda utilizar Python 3.7 o superior.
- La entrega se realizará a través de Aula en un solo archivo en formato `.zip` de nombre `T1_APELLIDO_ROL.zip` que incluya los siguientes archivos:
  - Un solo archivo `README.txt` con el nombre y ROL USM del estudiante, además de cualquier aclaración que sea necesaria.
  - Un solo archivo `.py` con el código fuente de la tarea.
  - Un solo archivo `.pdf` con el informe completo del desarrollo de la tarea. Se recomienda utilizar Overleaf (LaTeX) u otra variante de TeXpara redactar la tarea.
- El informe debe contener las siguientes secciones, cada una ordenada en páginas distintas y con toda la información necesaria.
  - Portada, incluyendo el nombre y rol del alumno, además de un título descriptivo.
  - Resumen, donde describa brevemente el desarrollo y resultados de la tarea.
  - Introducción, dejando claro el objetivo de la tarea y cualquier fórmula que utilice.
  - Desarrollo, explicando detalladamente la resolución de la tarea.
  - Resultados, con todos los resultados que haya obtenido durante el desarrollo de la tarea. Incluya extractos de cualquier prueba que haga con su programa.
  - Análisis, donde discuta los resultados de la sección anterior y cualquier complicación con la que se haya encontrado.
  - Conclusión, comentando el nivel de finalización de la tarea. Además explique generalmente la función de las distintas bases numéricas y códigos de corrección estudiados en clases.
- **Todas las preguntas respecto a la tarea deben hacerse a través del foro de consultas en Aula. No se responderán dudas durante las 48 horas previas a la entrega.**
- **La fecha límite de entrega de la tarea es el domingo 2 de mayo de 2021, a las 23:59 horas.**
- **Solo se recibirán tareas con un retraso máximo de 3 días (72 horas) desde la fecha límite de entrega, las cuales podrán optar a una nota máxima de 75. Cualquier retraso por sobre los 3 días mencionados será evaluado con nota 0.**
- Toda duda resuelta en el foro de consultas de Aula se tomará como un nuevo requisito evaluado para la entrega de la tarea. Redacte bien sus dudas para evitar confusiones.