

Midterm – 2 Sem 2014

1. *Pierre de Fermat*, famoso matemático francés del siglo XVII, demostró que un número primo p es expresable como suma de dos cuadrados si y sólo si $p = 2$ ó $p - 1$ es divisible por 4.

Por ejemplo:

$$29 = 2^2 + 5^2 = 4 \cdot 7 + 1 \quad 41 = 4^2 + 5^2 = 4 \cdot 10 + 1$$

En esta pregunta debes escribir un programa que pida al usuario un entero n e imprima los primeros n números primos que satisfacen esta propiedad, indicando cuál es su descomposición como suma de cuadrados. Por ejemplo, si se ingresa $n = 3$ tu programa debe mostrar:

```
> Ingrese un número: 3
> 2 = 1^2 + 1^2
> 5 = 1^2 + 2^2
> 13 = 2^2 + 3^2
```

2. En esta pregunta representaremos un vector tridimensional como una lista de tres números. Deberás crear un programa que calcule la posición y la rapidez de un objeto en un instante de tiempo t , aplicando la segunda ley de Newton. La fuerza (\vec{F}), la aceleración (\vec{a}), la posición inicial (\vec{r}_0), la posición (\vec{r}), y la velocidad (\vec{v}) son vectores tridimensionales.

Tu programa debe pedir al usuario la posición inicial, la fuerza aplicada al objeto, el tiempo en que se quiere realizar el cálculo y la masa del objeto, e imprimir como resultado las tres componentes del vector posición y el valor de la rapidez en el tiempo indicado. Debes suponer que el objeto está inicialmente en reposo.

Para evitar repetir código en tu programa, debes implementar y usar en él tres funciones:

- Multiplicar cada una de las componentes de un vector en \mathbb{R}^3 por un valor escalar.
- Sumar dos vectores en \mathbb{R}^3 , componente a componente.
- Calcular la norma de un vector en \mathbb{R}^3 (para el cálculo de la rapidez).

Las siguientes ecuaciones te pueden ayudar a resolver el problema:

$$\begin{aligned}\vec{F} &= m \cdot \vec{a} \\ \vec{v} &= \vec{v}_0 + \vec{a}t \\ \vec{r} &= \vec{r}_0 + \vec{v}_0t + \frac{1}{2}\vec{a}t^2 \\ \text{rapidez} &= \sqrt{v_x^2 + v_y^2 + v_z^2}, \text{ para } \vec{v} = (v_x, v_y, v_z)\end{aligned}$$

3. La distancia *Levenshtein* corresponde al menor número de caracteres que hay que insertar, borrar o sustituir para transformar un string a otro. Por ejemplo:

- Las palabras *gato* y *gatito* están a distancia 2, pues para llegar de una a otra basta insertar *i* e insertar *t* (o bien eliminar *i* y eliminar *t*).
- Las palabras *hola* y *ola* están a distancia 1; para llegar de una a otra basta insertar *h* (o borrar *h*).
- Las palabras *gallina* y *gallina* están a distancia 0, pues son iguales.
- Las palabras *caro* y *cara* están a distancia 1; para llegar de una a otra basta sustituir *o* por *a*.

En esta pregunta debes escribir un programa que pida dos strings al usuario e imprima si ellos están a distancia de Levenshtein 0, mayor que 1, o igual a 1. Si la distancia es igual a uno, se debe indicar la operación (insertar/borrar, o sustituir). A continuación se muestran tres diálogos que ejemplifican cómo debiese funcionar tu programa:

- > Palabra 1? jaron
> Palabra 2? jarron
> Respuesta: 1 operación (insertar/borrar)
- > Palabra 1? Limon
> Palabra 2? limon
> Respuesta: 1 operación (sustituir)
- > Palabra 1? jaron
> Palabra 2? melon
> Respuesta: más de 1 operación

