Requerimientos del Programa 5

Utilizando el **PSP 2.1** escribe un programa que:

- Lea del teclado dos datos:
  - *p* (número real entre 0 y 0.5)
  - *dof* (número entero mayor a cero)

- Calcule *x* tal que *p* se igual a *t(x, dof)*, o sea, encontrar el valor de *x* tal que al integrar de 0 a *x* la "*distribución t*" con *dof* grados de libertad nos de la *p* que se leyó. Para calcular la integral se utilizará el código del programa 4.

- Escriba en pantalla estos dos valores leídos y el valor calculado, de acuerdo con el siguiente formato:

```
p   = x.xxxxx
dof = xx
x   = x.xxxxx
```

NOTA:

✓ Los valores de *x* y *p* se desplegarán con 5 decimales (redondeados hacia arriba en su último dígito, por ejemplo: 0.123455 se desplegará como 0.12346, mientras que 0.123454 se desplegará como 0.12345)

Otras características que ***debe*** cumplir el programa:

- No utilizará ningún GUI para operar (funcionará desde la consola)
- Debe estar construido con programación orientada a objetos
- Debe contar con al menos 3 clases "relevantes" (la clase que contiene el "main" se cuenta como una de estas 3 clases)
- El ***<u>único</u>*** código que puede ser reutilizado es el de tus programas 1 a 4
- Debe manejar apropiadamente ***<u>todas</u>*** las condiciones normales y <u>anormales</u>
- Debe pasar exitosamente ***<u>todos</u>*** los casos de prueba (*error máximo 0.0001*):
  - Los diseñados por ti <u>en la fase de diseño</u>, y
  - Los siguientes 3 casos de prueba (es obligatorio incluirlos en el Diseño de las Pruebas):

Objetivo de la prueba	Instrucciones y datos de entrada	Resultados Esperados

Probar con datos correctos	Teclear en pantalla:
	0.2
	6
		`p = 0.20000`
		`dof = 6`
		`x = 0.55338`

Probar con datos correctos	Teclear en pantalla:
	0.45
	15
		`p = 0.45000`
		`dof = 15`
		`x = 1.75305`

Probar con datos correctos	Teclear en pantalla:
	0.495
	4
		`p = 0.49500`
		`dof = 4`
		`x = 4.60409`

Fin de los requerimientos

Explicación y ejemplo de cómo se realizan los cálculos (no son requerimientos)

(Tomado del curso original del PSP del Software Engineering Institute)

Program 5 algorithm

Finding the value of *x*

Find the value of x for which integrating the t function from 0 to x gives a result of p.

- Start with a trial value for upper limit of 1 and calculate the value of the integration.
- Compare it to the desired value.
  - if the result of the integration is too low, pick a larger trial upper limit
  - if the result of the integration is too high, pick a smaller trial upper limit

Make successive trial integrations until the value of the integration is within an acceptable error, for example 0.00000001.

One way to make this calculation is as follows.

Step	*Action*
1	Start with a trial value of x (for example, 1.0).
2.	Make an initial integral and test to see if it gives the proper value; if not, continue.
3.	If it is too low, add $d = 0.5$ to trial x.
4.	If it is too high, subtract $d = 0.5$ from trial x.
5.	Integrate again and test if the result (x) is within an acceptable error; if not, continue.
6.	If too low, adjust d (see below); add d to trial x.
7.	If too high, adjust d (see below); subtract d from trial x.
8.	Go to step 5.

The rules for adjusting d are these.

1. As long as the tests for the error of the result give the same sign of the error, leave d unchanged.
2. Whenever the sign of the error changes, divide d by 2.

Note that this method of adjusting d could result in a trial value of $x = 0$.

To guard against a problem with Simpson's method, ensure that the program will handle a 0 value of the function being integrated.

An example

An example

In this example, we'll calculate the value of x for which integrating the t function from 0 to x gives a result of $p=0.379$ with 5 degrees of freedom.

1. $P_t = 0.379$ (Target probability, provided as input)
2. $dof = 5$ (degrees of freedom, provided as input)
3. Set the maximum error, for example $E = 0.0000001$
4. Set the initial value of x, for example $x_0 = 1.0$
5. Set the initial delta increment, dividing x by 2: $d_0 = x_0 / 2 = 0.5$
6. Use program 4 to calculate $p_0 = t(x_0, dof) = t(1.0, 5) = 0.318391266$
7. Since $p_0 < P_t$, we need to move x to the right, so $x_1 = x_0 + d_0 = 1.5$
8. We use program 4 to calculate $p_1 = t(x_1, dof) = t(1.5, 5) = 0.403048160$
9. Since the difference between the last 2 calculations is greater than E, we continue.
10. Since $p_1 > P_t$, we need to move x to the left, but since there is a change of direction, we first divide d by 2, so $d_1 = d_0 / 2 = 0.25$ and $x_2 = x_1 - d_1 = 1.25$
11. We continue like this, until the difference between the last 2 calculations is less than E

The calculations look like this:

i	$x_i = x_{i-1} \pm d_{i-1}$	d_i	$p_i = t(x_i, dof)$	p_i vs P_t	Direction	$E_i = \| p_i - p_{i-1} \|$	$E_i < E$?
0	1.000000000	0.500000000	0.318391266	pi < 0.37900	Right	NA	NA
1	1.500000000	0.250000000	0.403048160	pi > 0.37900	Left	0.084656894	No... continue
2	1.250000000	0.125000000	0.366691889	pi < 0.37900	Right	0.036356271	No... continue
3	1.375000000	0.062500000	0.386226095	pi > 0.37900	Left	0.019534207	No... continue
4	1.312500000	0.031250000	0.376816254	pi < 0.37900	Right	0.009409841	No... continue
5	1.343750000	0.015625000	0.381608249	pi > 0.37900	Left	0.004791994	No... continue
6	1.328125000	0.015625000	0.379234303	pi > 0.37900	Left	0.002373945	No... continue
7	1.312500000	0.007812500	0.376816254	pi < 0.37900	Right	0.002418049	No... continue
8	1.320312500	0.007812500	0.378030827	pi < 0.37900	Right	0.001214573	No... continue
9	1.328125000	0.003906250	0.379234303	pi > 0.37900	Left	0.001203476	No... continue
10	1.324218750	0.001953125	0.378633948	pi < 0.37900	Right	0.000600356	No... continue
11	1.326171875	0.001953125	0.378934471	pi < 0.37900	Right	0.000300523	No... continue
12	1.328125000	0.000976563	0.379234303	pi > 0.37900	Left	0.000299833	No... continue
13	1.327148438	0.000976563	0.379084473	pi > 0.37900	Left	0.000149830	No... continue
14	1.326171875	0.000488281	0.378934471	pi < 0.37900	Right	0.000150003	No... continue
15	1.326660156	0.000244141	0.379009494	pi > 0.37900	Left	0.000075023	No... continue
16	1.326416016	0.000122070	0.378971988	pi < 0.37900	Right	0.000037506	No... continue
17	1.326538086	0.000122070	0.378990742	pi < 0.37900	Right	0.000018754	No... continue
18	1.326660156	0.000061035	0.379009494	pi > 0.37900	Left	0.000018752	No... continue
19	1.326599121	0.000061035	0.379000118	pi > 0.37900	Left	0.000009375	No... continue
20	1.326538086	0.000030518	0.378990742	pi < 0.37900	Right	0.000009376	No... continue
21	1.326568604	0.000030518	0.378995430	pi < 0.37900	Right	0.000004688	No... continue
22	1.326599121	0.000015259	0.379000118	pi > 0.37900	Left	0.000004688	No... continue
23	1.326583862	0.000007629	0.378997774	pi < 0.37900	Right	0.000002344	No... continue
24	1.326591492	0.000007629	0.378998946	pi < 0.37900	Right	0.000001172	No... continue
25	1.326599121	0.000003815	0.379000118	pi > 0.37900	Left	0.000001172	No... continue
26	1.326595306	0.000001907	0.378999532	pi < 0.37900	Right	0.000000586	No... continue
27	1.326597214	0.000001907	0.378999825	pi < 0.37900	Right	0.000000293	No... continue
28	1.326599121	0.000000954	0.379000118	pi > 0.37900	Left	0.000000293	No... continue
29	1.326598167	0.000000477	0.378999972	pi < 0.37900	Right	0.000000146	No... continue
30	1.326598644	0.000000238	0.379000045	pi > 0.37900	Left	0.000000073	Yes... stop

We return the last x result, $x_{30} = 1.326598644$