Universidad de Los Andes
Núcleo Universitario “Alberto Adriani”
Programación Digital 10
Prof. Lisbeth Pérez

Ejercicios Repetición a enviar el día 28 de mayo de 2011.
Hora tope: 11.59pm

Ejercicio 1:

Escribir un programa que diga si un número entero es primo o no.

Análisis E-P-S:
Entrada:
El número que deseamos saber si es primo o no.

Descripción	Identificador	Tipo	Condición
El número	num	entero	ninguna

Proceso:

Los números primos son aquellos que son divisibles solo por el 1 y por sí mismos.

Nuestro programa deberá consistir en calcular los divisores del número dado por el usuario; si el número de divisores es 2, estaremos en presencia de un número primo, de lo contrario no lo será.
La mejor forma para buscar los divisores es comparando los residuos de la división del numero (num) dado por el usuario con todos los números en el intervalo [1,num], si nos encontramos con un residuo cero, estamos en la presencia de un divisor y contamos.

Debemos tener un contador para los divisores y un contador para movernos en el intervalo [1,num]

Descripción	Identificador	Tipo	Condición
Contador de divisores	divisor	entero	ninguna
Contador entre 1 y num	contador	Entero	ninguna

Haremos la siguiente operación

residuo(num/1) --> si es 0, incremento el contador de divisores
residuo(num/2) --> si es 0, incremento el contador de divisores
residuo(num/3) --> si es 0, incremento el contador de divisores

...
residuo(num/num)--> si es 0, incremento el contador de divisores

la operación que se repite es

residuo(num/?), --> si es 0, incremento el contador de divisores

donde:
? = variación entre 1 y num (contador)
?=contador

por tanto, la operación final es:
residuo(num/contador) --> si es 0, divisor=divisor+1

Si al finalizar el proceso, el contador de divisores es igual a 2 entonces el número es primo (solo encontró dos divisores), de lo contrario no es primo.

Salida: Un mensaje por pantalla indicando si el número es primo o no.

Algoritmo con repita mientras:

1. inicio
2. divisor=0
3. contador=1
4. Escribir("Inserte el numero que desea verificar : ")
5. Leer(num)
6. Mientras(contador<=num)
   6.1 Si(residuo(num/contador)=0) entonces
   6.1.1 divisor=divisor+1
   Fin_si
   6.2 contador=contador+1
   Fin_RM
7. Si(divisor=2) entonces
   7.1 Escribir("el numero es primo")
   sino
   7.2 Escribir("el numero no es primo")
   fin_si
8. Fin

Codificación

```
#include<iostream>
using namespace std;
int main(){
	//declaramos variables y aprovechamos para inicializar
	int num,contador=1,divisor=0;
	cout<<"Inserte el numero que desea verificar: ";
	cin>>num;
	while(contador<=num){
		if(num%contador==0)
			divisor=divisor+1;
		contador=contador+1;
	}
	if(divisor==2)
		cout<<"\nEl numero es primo"<<endl;
	else
		cout<<"\nEl numero no es primo"<<endl;
	return 0;

}
```

Algoritmo con Repita para: Observen que tanto la inicialización como la modificación del contador se hacen dentro de la sentencia Para

1. inicio
2. divisor=0
3. Escribir(“Inserte el numero que desea verificar : “)
4. Leer(num)
5. Para(contador=1;contador<=num;contador=contador+1)
   5.1 Si(residuo(num/contador)=0) entonces
   5.1.1 divisor=divisor+1
   Fin_si
   Fin_RP
6. Si(divisor=2) entonces
   6.1 Escribir(“el numero es primo”)
   sino
   6.2 Escribir(“el numero no es primo”)
   fin_si
7. Fin

Codigo con Repita para:

```
#include<iostream>
using namespace std;
int main(){
	//declaramos variables
	int num,contador,divisor=0;
	cout<<"Inserte el numero que desea verificar: ";
	cin>>num;
	for(contador=1;contador<=num;contador++)
		if(num%contador==0)
			divisor=divisor+1;

	if(divisor==2)
		cout<<"\nEl numero es primo"<<endl;
	else
		cout<<"\nEl numero no es primo"<<endl;
	return 0;
}
```

Ambos algoritmos funcionan, sin embargo notese que no tiene sentido seguir recorriendo el intervalo 1-num, luego que el número de divisores es superior a 2. Ejemplo: Si estamos verificando el numero 100, nuestro algoritmo funciona de la siguiente manera:

contador	Condicion contador<=100?	100%contador==0?	divisor
1	1<=100? Verdadero	100%1==0? V	0+1=1
2	2<=100? V	100%2==0?V	0+1+1=2
3	3<=100? V	100%3==0?F	
4	4<=100? V	100%4==0? V	0+1+1+1=3
5	5<=100? V	100%5==0? V	0+1+1+1+1=4
....	....	....	...
100	100<=100? V	100%100==0? V	divisor+1

Como puede observarse, desde la cuarta iteración ya sabemos que nuestro número NO ES PRIMO, y estamos gastando tiempo y recursos al continuar ejecutando el código. En realidad, solo deberiamos ejecutar el código para un contador entre 1 y num y el divisor<3.

1. inicio
2. divisor=0
3. Escribir(“Inserte el numero que desea verificar : “)
4. Leer(num)
5. Para(contador=1;contador<=num y divisor<3;contador=contador+1)

5.1 Si(residuo(num/contador)=0) entonces
5.1.1 divisor=divisor+1
Fin_si
Fin_RP
6. Si(divisor=2) entonces
6.1 Escribir(“el numero es primo”)
sino
6.2 Escribir(“el numero no es primo”)
fin_si
7. Fin

Codigo con Repita para:

```
#include<iostream>
using namespace std;
int main(){
	//declaramos variables
	int num,contador,divisor=0;
	cout<<"Inserte el numero que desea verificar: ";
	cin>>num;
	for(contador=1;contador<=num && divisor<3;contador++)
		if(num%contador==0)
			divisor=divisor+1;

	if(divisor==2)
		cout<<"\nEl numero es primo"<<endl;
	else
		cout<<"\nEl numero no es primo"<<endl;
	return 0;
}
```

Veamos la corrida en frio:

contador	Condicion contador<=100 y divisor<3?	100%contador==0?	divisor
1	1<=100 y 0<3? V y V=V	100%1==0? V	0+1=1
2	2<=100 y 1<3? V y V = V	100%2==0?V	0+1+1=2
3	3<=100 y 2<3? V y V = V	100%3==0?F	
4	4<=100 y 2<3? V y V=V	100%4==0? V	0+1+1+1=3
5	5<=100 y 3<3? V y F= F	FIN DEL LAZO	

Hemos mejorado notoriamente el programa.

Podemos refinarlo aún más: Ya sabemos que 1 y num SIEMPRE serán divisores de cualquier número, por tanto tampoco tiene sentido que realizar esta iteración.

Actividad 1:
Modifique el programa para que el contador varie entre 2 y num-1, observe que deberá cambiar la condicion de los divisores; cuantos divisores tendremos ahora cuando el número sea primo?

Nota: Pueden copiar y pegar el código de este documento y agregar o modificar las líneas que sean necesarias.

Actividad 2:

Que debemos hacer si ahora queremos verificar números mientras el usuario así lo desee? Es decir, si queremos preguntarle al usuario si desea verificar otro número, y en caso de respuesta afirmativa realizar el procedimiento nuevamente? Nota: Recuerde la estructura hacer-mientras y agregue las líneas necesarias al código.

Actividad 3:

Realice un programa que determine si un número es perfecto.

Un **número perfecto** es un número natural que es igual a la suma de sus divisores, sin incluirse él mismo.

Ejemplo: 6 es un número perfecto, porque sus divisores propios son 1, 2 y 3; y 1 + 2 + 3 =6. Los siguientes números perfectos son 28, 496 y 8128.