Universidad Nacional de Ingeniería
Facultad de Ciencias
Escuela Profesional de Ciencia de la Computación
Introducción a la Programación

CC102 **Ciclo: 2017-1**

Solución Examen parcial 8/5/2017

1. [5 ptos.] La conjetura de Golbach es uno de los problemas sin resolver más conocidos de la teoría de números. Establece que todo entero par mayor que 2 puede expresarse como la suma de dos primos. Implemente un programa que verique la validez de dicha conjetura hasta 100 imprimiendo sólo una descomposición de n como suma de dos primos para n = 4, 6, . . . , 100

```
#include<stdio.h>
int primo(int n); // Prototipo de la función primo
int golbach(int n); // Prototipo de la función golbach
void main(){
        int n = 4;
        for ( ; n <= 100 ; n = n + 2 ) golbach(n);
}

int primo(int n) {// cabecera de la definición de la función primo
        int d=2;
        for ( ; d < n ; d++)
                if (n%d == 0) return 0; // esto ocurre cuando hay un divisor propio de n
        return 1; // retorna 1 cuando n es primo y 0 caso contrario
}

int golbach(int n) {// cabecera de la definición de la función golbach
        int a,b; // a estas variables las iremos asignando todas las posibles a,b cuya suma sea n
        int pri;
        int ok = 0;
        for( a = 2 ; a < n ; a++ ){
                for( b = a ; b < n ; b++ ){
                        pri = primo(a)*primo(b); // pri = 1 si y solo si a y b son primos
                        if ( (pri == 1) && (n == a + b) ) {
                                printf(" %d = %d + %d.\n",n,a,b);
                                return 0;
                        }
                }
        }
}
```

2. [5 ptos.] Escriba un programa que pida ingresar el numerador y el denominador de una fracción. Luego, se debe mostrar su equivalente irreducible. Por ejemplo, si se ingresa 4 como numerador y 6 como denominador, se debe mostrar 2/3. Se debe emplear al menos una función definida por el usuario.

```
#include <stdio.h>
int     mcd (int m, int n);
void    reducir (int m, int n);
void main (void){
        int M, N;
```

```
	printf (" Ingrese el numerador y denominador de una fracción: ");
	scanf ("%d %d", &M, &N);
	reducir (M,N);
}

int mcd (int m, int n){
	int aux, r;
	if (m < n){
		aux = m;
		m = n;
		n = aux;
	}
	r = m%n;
	while (r != 0){
		m = n;
		n = r;
		r = m%n;
	}
	return n;
}

void	reducir (int m, int n){
	int MCD = mcd(m,n);
	printf (" La fracción reducida es %d/%d\n", m/MCD, n/MCD);
}
```

3. [5 ptos.] Escriba un programa que: lea un entero **n** > 0, genere un arreglo de longitud **n** con números aleatorios entre **10** y **20**, lo imprima, lo ordene por pares e impares, y lo imprima.
   Sugerencia, para facilitarle la vida, use dos funciones:
   - crear(…) // imprima allí mismo
   - ordenar(…) // Apóyese en otro arreglo auxiliar. Imprima allí mismo

```
// Ordenar arreglo por pares e impares
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void crear  (int arr[], int n);
void ordenar(int arr[], int n);
void mostrar(int arr[], int n);

void main(void){
    int n;
    printf("Ingrese un entero > 1: ");
    scanf("%d", &n);
    int arr[n];

    crear  (arr, n);
    printf("Arreglo original:\n");
    mostrar(arr, n);

    ordenar(arr, n);
    printf("Arreglo ordenado:\n");
    mostrar(arr, n);
}
```

```
void crear(int arr[], int n){
    int i;
    srand(time(NULL));
    for(i=0; i<n; i++) arr[i] = rand()%11+10;
}

void ordenar(int arr[], int n){
    int i, j = 0, k= n-1, aux[n];
    for(i=0; i<n; i++)
            if(arr[i]%2==0) aux[j++] = arr[i];
            else                    aux[k--] = arr[i];
    for(i=0; i<n; i++) arr[i]   = aux[i];
}

void mostrar(int arr[], int n){
    int i;
    for(i=0; i<n; i++) printf("%d\t", arr[i]);
    printf("\n");
}
```

4. [5 ptos.] Sea el arreglo arr[3][3] = {9, 8, 7, 6, 5, 4, 3, 2, 1}. Escriba un programa para mostrarlo en pantalla, ordenarlo ascendentemente por la columna 0 y mostrarlo nuevamente.

```
#include <stdio.h>
void mostrar(int arr[][3], int m, int n){
        int i, j;
        for(i=0; i< m; i++){
                for(j=0; j< n; j++) printf("%d\t", arr[i][j]);
                printf("\n");
        }
}
void main(void){
        int n = 3, arr[3][3] = {9, 8, 7, 6, 5, 4, 3, 2, 1}, i, j, k, aux;
        printf("Arreglo original:\n");
        mostrar(arr, 3, 3);

        i = 0;                                          // ordenar el arreglo
        while(i<n){
                j=n-1;
                while(j>0){
                        if(arr[j][0] < arr[j-1][0])
                                for (k=0; k<3; k++) {
                                        aux         = arr[j][k];
                                        arr[j][k]  = arr[j-1][k];
                                        arr[j-1][k]= aux;
                                }
                        j=j-1;
                }
                i=i+1;
        }
        printf("\nArreglo ordenado:\n");
        mostrar(arr, 3, 3);
}
```