Universidad Autónoma de Baja California

Facultad de Ingeniería Arquitectura y Diseño

Ingeniero en Software y tecnologías emergentes

Materia: Programación Estructurada / Clave **36276**

Alumno: Antonio

Matrícula: 372576

Maestro: Pedro Núñez Yépiz

Actividad No. 9-3/4 : CURP (ANEXO)

Tema – Unidad 5 : arreglos y Cadenas

Ensenada Baja California a 22 de octubre del 2023

```
// Antonio Ramos Gonzalez Mt: 372576
// 10/12/2023 || 10/22/2023
// se pediran datos al usuario para poder generar su CURP, haciendo todas las validaciones del mismo
// RGA_Act9.75_932

#include "Babilonia.h"

int msg();
void menu();
void curp();
int nombre(char curp[], char nomb[], char nomb2[], char appP[], char apM[]);
int nacimiento(char curp[]);
void sexo(char curp[]);
void estados(char curp[]);
void imprimirEstados();
void consonantes(char curp[], char nomb[], char nomb2[], char appP[], char apM[], int val);
void generacio(char curp[], int year);
void numAl(char curp[]);
int nomb_Novalid(char nomb[]);
bool palabrasInconvenientes(char curp[]);
void validConsoAp(char curp[], char cad[]);
void validConsoNomb(char curp[], char cad[]);
void validnumb(char cad[]);
void digitos(char curp[], char cad[]);
void caractapP(char curp[], char cad[]);
void caractapM(char curp[], char cad[]);
void caractano(char curp[], char cad[]);
void conv(char cad[]);

int main()
{
    srand(time(NULL));
    fflush(stdin);
    menu();
    return 0;
}

int msg()
{
    printf("MENU\n");
    printf("1.-Generar curp\n");
    printf("0.-Salir\n");
    return valid("Ingresa una opcion: ", 0, 1);
}

```

```
void menu()
{
    int opc;
    do
    {
        system("CLS");
        opc = msg();
        switch (opc)
        {
        case 1:
            curp();
            break;
        }
    } while (opc != 0);
}

void curp()
{
    char curp[18], falap[10], falam[10], falno[10], falno2[10];
    char nomb[30], nomb2[30], apP[30], apM[30];
    int year, val, inc;
    curp[0] = '\0';
    do
    {
        validCad("Ingrese nombre: ", nomb);

    } while (nomb[0] == '\0' || strlen(nomb) > 15);

    do
    {
        validCad("Ingrese 2do nombre(si tiene mas de 2 ingreselos aqui): ", nomb2);
    } while (strlen(nomb2) > 30);

    do
    {
        validCad("Ingrese apellido paterno: ", apP);
    } while (strlen(apP) > 15);

    do
    {
        validCad("Ingrese apellido materno: ", apM);
    } while (strlen(apM) > 15);

    strcpy(falap, apP);
    validnumb(falap);
    strcpy(falam, apM);
    validnumb(falam);
    strcpy(falno, nomb);
```

```
    validnumb(falno);
    strcpy(falno2, nomb2);
    validnumb(falno2);

    val = nombre(curp, nomb, nomb2, apP, apM);

    year = nacimiento(curp);

    sexo(curp);

    estados(curp);

    consonantes(curp, nomb, nomb2, apP, apM, val);

    generacio(curp, year);

    numAl(curp);

    inc = palabrasInconvenientes(curp);

    caractapP(curp, falap);
    caractapM(curp, falam);
    if (nomb2[0] == '\0')
    {
        caractano(curp, falno);
    }
    else
    {
        if (val == 0)
        {
            caractano(curp, falno);
        }
        else
        {
            caractano(curp, falno2);
        }
    }

    if (inc == 1)···
    printf("Su curp es: %s\n", curp);
    system("PAUSE");
}

int nombre(char curp[], char nomb[], char nomb2[], char apP[], char apM[])
{
    int val;

    if (apP[0] == '\0')
```

```
{
    int val;

    if (apP[0] == '\0')
    {
        apP[0] = 'X';
        apP[1] = 'X';
        apP[2] = '\0';
    }
    else
    {
        validnumb(apP);
    }
    conv(apP);
    strncat(curp, apP, 2);
    digitos(curp, apP);

    curp[2] = '\0';

    if (apM[0] == '\0')
    {
        apM[0] = 'X';
        apM[1] = 'X';
        apM[2] = '\0';
    }
    else
    {
        validnumb(apM);
    }
    conv(apM);
    strncat(curp, apM, 1);
    curp[3] = '\0';

    val = nomb_Novalid(nomb);

    if (nomb2[0] == '\0')
    {

        validnumb(nomb);
        conv(nomb);
        strncat(curp, nomb, 1);
        curp[4] = '\0';
    }
    else
    {
        if (val == 0)
        {
            validnumb(nomb);
            conv(nomb);
```

```
            conv(nomb2);
            strncat(curp, nomb2, 1);
            curp[4] = '\0';
        }
    }

    return val;
}

int nacimiento(char curp[])
{
    char anio[5], mes[3], dia[3];
    int year, mont, day, lar, bis;

    system("CLS");
    year = valid("Ingresa anio de nacimiento: ", 1950, 2023);
    if (year == 2023)
    {
        mont = valid("Ingresa mes de nacimiento(enero=01,diciembre=12): ", 1, 10);
    }
    else
    {
        mont = valid("Ingresa mes de nacimiento(enero=01,diciembre=12): ", 1, 12);
    }
    if (mont == 1 || mont == 3 || mont == 5 || mont == 7 || mont == 8 || mont == 10 || mont == 12)
    {
        day = valid("Ingresa dia de nacimiento: ", 1, 31);
    }
    if (mont == 4 || mont == 6 || mont == 9 || mont == 11)
    {
        day = valid("Ingresa dia de nacimiento: ", 1, 30);
    }
    if (mont == 2)
    {
        bis = bisiesto(year);
        if (!bis)
        {
            day = valid("Ingresa dia de nacimiento: ", 1, 28);
        }
        else
        {
            day = valid("Ingresa dia de nacimiento: ", 1, 29);
        }
    }

    snprintf(anio, 5, "%d", year);
    snprintf(mes, 3, "%02d", mont);
    snprintf(dia, 3, "%02d", day);
```

```
    lar = strlen(anio);
    anio[0] = anio[lar - 2];
    anio[1] = anio[lar - 1];

    strncat(curp, anio, 2);
    curp[6] = '\0';
    strncat(curp, mes, 2);
    curp[8] = '\0';
    strncat(curp, dia, 2);
    curp[10] = '\0';

    return year;
}

void sexo(char curp[])
{
    char sex[2];
    int s;
    s = valid("ingresa sexo(1.-Hombre, 2.-Mujer): ", 1, 2);

    if (s == 1)
    {
        strcpy(sex, "H");
        strncat(curp, sex, 1);
    }
    else
    {
        strcpy(sex, "M");
        strncat(curp, sex, 1);
    }

    curp[11] = '\0';
}

void estados(char curp[])
{
    char est[2];
    int a;
    char estados[33][3] = {
        "AS", "BC", "BS", "CC", "CS", "CH", "CL", "CM", "DG",
        "GT", "GR", "HG", "JC", "MC", "MN", "MS", "NT", "NL",
        "OC", "PL", "QT", "QR", "SP", "SL", "SR", "TC", "TS",
        "TL", "VZ", "YN", "ZS", "DF", "NE"};

    imprimirEstados();

    a = valid("Ingrese estado: ", 1, 33);
    a--;
```

```
    a--;
    strcpy(est, estados[a]);
    strncat(curp, est, 2);
    curp[13] = '\0';
}

void imprimirEstados()
{
    int i;
    char estados[33][30] = {
        "Aguascalientes",
        "Baja California",
        "Baja California Sur",
        "Campeche",
        "Chiapas",
        "Chihuahua",
        "Coahuila",
        "Colima",
        "Durango",
        "Guanajuato",
        "Guerrero",
        "Hidalgo",
        "Jalisco",
        "Estado de Mexico",
        "Michoacan",
        "Morelos",
        "Nayarit",
        "Nuevo Leon",
        "Oaxaca",
        "Puebla",
        "Queretaro",
        "Quintana Roo",
        "San Luis Potosi",
        "Sinaloa",
        "Sonora",
        "Tabasco",
        "Tamaulipas",
        "Tlaxcala",
        "Veracruz",
        "Yucatan",
        "Zacatecas",
        "Ciudad de Mexico",
        "Extranjero"};

    for (i = 0; i < 33; i++)
    {
        printf("%d.- %s\n", i + 1, estados[i]);
    }
```

```c
void consonantes(char curp[], char nomb[], char nomb2[], char apP[], char apM[], int val)
{
    char conso_nomb[10], conso_apP[10], conso_apM[10];
    int lar_cad;
    /*************************************************************************/
    lar_cad = strlen(apP);
    solo_consonantes(apP, lar_cad, conso_apP);

    validConsoAp(curp, conso_apP);
    conv(apP);
    curp[14] = '\0';
    /*************************************************************************/
    lar_cad = strlen(apM);
    solo_consonantes(apM, lar_cad, conso_apM);

    validConsoAp(curp, conso_apM);

    curp[15] = '\0';
    /*************************************************************************/
    if (nomb2[0] == '\0')
    {
        lar_cad = strlen(nomb);
        solo_consonantes(nomb, lar_cad, conso_nomb);
        validConsoNomb(curp, conso_nomb);
    }
    else
    {
        if (val == 0)
        {
            lar_cad = strlen(nomb);
            solo_consonantes(nomb, lar_cad, conso_nomb);
            validConsoNomb(curp, conso_nomb);
        }
        else
        {
            lar_cad = strlen(nomb2);
            solo_consonantes(nomb2, lar_cad, conso_nomb);
            validConsoNomb(curp, conso_nomb);
        }
    }
    curp[16] = '\0';
}

void generacio(char curp[], int year)
{
    char gen[2];
    if (year < 2000)
    {
```

```
    char gen[2];
    if (year < 2000)
    {
        strcpy(gen, "0");
        strncat(curp, gen, 2);
    }
    else
    {
        if (year < 2010)
        {
            strcpy(gen, "A");
            strncat(curp, gen, 2);
        }
        else
        {
            if (year < 2020)
            {
                strcpy(gen, "B");
                strncat(curp, gen, 2);
            }
            else
            {
                strcpy(gen, "C");
                strncat(curp, gen, 2);
            }
        }
    }
    curp[18] = '\0';
}

void numAl(char curp[])
{
    int num, i;
    char n[2];
    for (i = 0; i < 10; i++)
    {
        num = rand() % 10;
    }
    snprintf(n, 2, "%d", num);

    strncat(curp, n, 2);
}

int nomb_Novalid(char nomb[])
{
    int i, band;
    char noVali[8][10] = {
        "MARIA",
```

```
bool palabrasInconvenientes(char curp[])
{
    char inc[80][5] =
        {
            "BAKA",
            "BUEI",
            "BUEY",
            "CACA",
            "CACO",
            "CAGA",
            "CAGO",
            "CAKA",
            "CAKO",
            "COGE",
            "COGI",
            "COJA",
            "COJE",
            "COJI",
            "COJO",
            "COLA",
            "CULO",
            "FALO",
            "FETO",
            "GETA",
            "GUEI",
            "GUEY",
            "JETA",
            "JOTO",
            "KACA",
            "KACO",
            "KAGA",
            "KAGO",
            "KAKA",
            "KAKO",
            "KOGE",
            "KOGI",
            "KOJA",
            "KOJE",
            "KOJI",
            "KOJO",
            "KOLA",
            "KULO",
            "LILO",
            "LOCA",
            "LOCO",
            "LOKA",
            "LOKO",
            "MAME",
```

```
            "MAME",
            "MAMO",
            "MEAR",
            "MEAS",
            "MEON",
            "MIAR",
            "MION",
            "MOCO",
            "MOKO",
            "MULA",
            "MULO",
            "NACA",
            "NACO",
            "PEDA",
            "PEDO",
            "PENE",
            "PIPI",
            "PITO",
            "POPO",
            "PUTA",
            "PUTO",
            "QULO",
            "RATA",
            "ROBA",
            "ROBE",
            "ROBO",
            "RUIN",
            "SENO",
            "TETA",
            "VACA",
            "VAGA",
            "VAGO",
            "VAKA",
            "VUEI",
            "VUEY",
            "WUEI",
            "WUEY"};
    char cpy[2];
    int i, ne;
    strncat(cpy, curp, 4);
    for (i = 0; i < 80; i++)
    {
        ne = strcmp(cpy, inc[i]);
        if (ne == 0)
        {
            return 1;
        }
    }
    return 0;
```

```
void validConsoAp(char curp[], char cad[])
{
    int len;
    char car[2];
    len = strlen(cad);
    if (len == 1)
    {
        strncat(curp, cad, 1);
    }
    else
    {
        if (cad[0] == curp[0])
        {
            car[0] = cad[1];
            strncat(curp, car, 2);
        }
        else
        {
            if (cad[0] == curp[2])
            {
                car[0] = cad[1];
                strncat(curp, car, 2);
            }
            else
            {
                car[0] = cad[0];
                strncat(curp, car, 2);
            }
        }
    }
}

void validConsoNomb(char curp[], char cad[])
{
    int len;
    char caract[2];
    len = strlen(cad);
    if (len == 1)
    {
        strncat(curp, cad, 1);
    }
    if (cad[0] == curp[3])
    {
        caract[0] = cad[1];
        strncat(curp, caract, 1);
    }
    else
```

```
    {
        caract[0] = cad[0];
        strncat(curp, caract, 1);
    }
}

void validnumb(char cad[])
{
    int len, i, j;
    len = strlen(cad);
    if (cad[1] == ' ')
    {
        for (j = 2, i = 0; i < len; i++, j++)
        {
            cad[i] = cad[j];
        }
        cad[i + 1] = '\0';
    }
    if (cad[2] == ' ')
    {

        if (cad[5] == ' ')
        {
            for (j = 6, i = 0; i < len; i++, j++)
            {
                cad[i] = cad[j];
            }
            cad[i + 1] = '\0';
        }
        else
        {

            for (j = 3, i = 0; i < len; i++, j++)
            {
                cad[i] = cad[j];
            }
            cad[i + 1] = '\0';
        }
    }
    if (cad[3] == ' ')
    {

        for (j = 4, i = 0; i < len; i++, j++)
        {
            cad[i] = cad[j];
        }
        cad[i + 1] = '\0';
    }
```

```
    }
}

void digitos(char curp[], char cad[])
{
    int len;
    char cons[15], voc[15];
    len = strlen(cad);
    solo_consonantes(cad, len, cons);
    solo_vocales(cad, len, voc);
    if (curp[0] == 'A' || curp[0] == 'E' || curp[0] == 'I' || curp[0] == 'O' || curp[0] == 'U')
    {
        curp[1] = voc[1];
    }

    if (curp[1] != 'A' && curp[1] != 'E' && curp[1] != 'I' && curp[1] != 'O' && curp[1] != 'U')
    {
        if (curp[0] == 'A' || curp[1] == 'E' || curp[1] == 'I' || curp[1] == 'O' || curp[1] == 'U')
        {
            curp[1] = voc[1];
        }
        else
        {
            curp[1] = voc[0];
        }
    }
}

void caractapP(char curp[], char cad[])
{
    char fal[8];
    int len;
    len = strlen(cad);

    if (!isalpha(cad[0]))
    {
        curp[0] = 'X';
    }
    if (!isalpha(cad[1]))
    {
        if (cad[1] != ' ')
        {
            curp[1] = 'X';
        }
    }

    solo_consonantes(cad, len, fal);
    len = strlen(fal);
    conv(fal);
```

```
    conv(fal);
    if (len != 1)
    {
        if (curp[1] == fal[1])
        {
            curp[13] = fal[2];
        }
    }
}

void caractapM(char curp[], char cad[])
{
    char fal[8];
    int len;
    len = strlen(cad);
    if (!isalpha(cad[0]))
    {
        if (cad[0] != ' ')
        {
            curp[2] = 'X';
        }
    }

    solo_consonantes(cad, len, fal);
    len = strlen(fal);
    conv(fal);
    if (len != 1)
    {
        if (curp[2] == fal[0])
        {
            curp[14] = fal[1];
        }
    }
}

void caractano(char curp[], char cad[])
{
    char fal[8];
    int len;
    len = strlen(cad);
    if (!isalpha(cad[0]))
    {
        if (cad[0] != ' ')
        {
            curp[3] = 'X';
        }
    }
```

```
        }
    }

    solo_consonantes(cad, len, fal);
    conv(fal);

    if (curp[3] == fal[0])
    {
        curp[15] = fal[1];
    }
}

void conv(char cad[])
{
    int i, len;
    len = strlen(cad);

    for (i = 0; i < len; i++)
    {
        if (!isalpha(cad[i]))
        {
            if (cad[i] != ' ')
            {
                cad[i] = 'X';
            }
        }
    }
}
```

```
12  void menu();
13  void curp();
14  int nombre(char curp[], char nomb[], char nomb2[], cha
15  int nacimiento(char curp[]);
16  void sexo(char curp[]);
17  void estados(char curp[]);
18  void imprimirEstados();
19  void consonantes(char curp[], char nomb[], char nomb2[
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Su curp es: FXLO970131HGRRPM05
Presione una tecla para continuar . . .
```

```
15    void menu();
16    void curp();
17    int nombre(char curp[], char nomb[], char nomb2[], char
18    int nacimiento(char curp[]);
19    void sexo(char curp[]);
20    void estados(char curp[]);
21    void imprimirEstados();
22    void consonantes(char curp[], char nomb[], char nomb2[]
23    void generacio(char curp[], int year);
24    void numAl(char curp[]);
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Su curp es: CAXC020328HNEHXWA3
Presione una tecla para continuar . . .
```

```
15  #include "Babilonia.h"
16
17  int msg();
18  void menu();
19  void curp();
20  int nombre(char curp[], char nomb[], char
21  int nacimiento(char curp[]);
22  void sexo(char curp[]);
23  void estados(char curp[]);
24  void imprimirEstados();
25  void consonantes(char curp[], char nomb[],
26  void generacio(char curp[], int year);
27  void numAl(char curp[]);
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Su curp es: LOXA071228HYNMXNA1
Presione una tecla para continuar . . .
```