Universidad Autónoma de Baja California

Facultad de Ingeniería, Arquitectura y Diseño

Ingenieria en Computacion

Programación Estructurara/36276

Santos Tirado Martin/00369705 Pedro

Nuñez Yepiz

Actividad N. 9 3/4

CURP

Ensenada Baja California, 23 de octubre del 2023

Introducción:

El alumno aprenderá a crear por su cuenta el CURP.

Competencia:

El alumno llevará a cabo los ejercicios planteados con ayuda del profesor que dejo en un documento sobre el CURP.

Fundamentos:

El alumno aprenderá a manejar y acostumbrarse a los códigos largos y complicados con la ayuda de funciones, vectores, matrices, librerías propias, entre otras; también, el alumno desarrollara el CURP paso a paso y que este quede bien.

Procedimiento:

CURP

CADENAS.

INSTRUCCIONES:

1.- APLICA TODASLAS FUNCIONES QUE REALIZASTE EN LA ACTIVIDAD 7 PARTE 2.

2.-REALIZA REPORTE DE PRACTICA

3.- SUBE TODOS LOS ARCHIVOS (REPORTE PRACTICA Y PROGRAMA)

Realiza un programa que sirva para generar el **CURP** de una persona.

El programa debe pedir los datos al usuario, generar, almacenar en una cadena y desplegar el CURP.

El programa deberá repetirse cuantas veces desee el usuario

Nota: el programa deberá estar 100% validado, de datos entrada así como las reglas que deben cumplir al generar el CURP.

Resultados y Conclusiones:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include "guasa.h"

const char *ent[] = {
"Aguascalientes", "Baja California", "Baja California Sur", "Campeche",
"Coahulia", "Colima", "Chiapas", "Chihuahua", "Distrito Federal", "Durango",
"Guanajuato", "Guerrero", "Hidalgo", "Jalisco", "Estado de mexico", "Michoaca",
"Morelos", "Nayarit", "Nuevo Leon", "Oaxaca", "Pueblo", "Queretaro", "Quintana
Roo", "San Luis Potosi", "Sinaloa", "Sonora", "Tabasco", "Tamaulipas",
"Tlaxcala", "Veracruz", "Yucatan", "Zacatecas", "Nacido en el extranjero"};
const char *ent_abrev[] = {
    "AS", "BC", "BS", "CC", "CL", "CM", "CS", "CH", "DF", "DG", "GT", "GR", "HG",
"JC", "MC", "MN", "MS", "NT", "NL", "OC", "PL", "QT", "QR", "SP", "SL", "SR",
"TC", "TS", "TL", "VZ", "YN", "ZS", "NE"};
const char *palabras_groseras[] = {
    "BACA", "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", "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"};
//*************************************************************************
*************************************************************************
**********//
int msgs();
void menu();
void curp();
int meses(char anio[], char mes[]);
void groserias(char cadena[]);
char gen(char cadena[]);
void entidad(char cadena[], char curp[]);
int jose_maria(char nombre1[], char nombre2[]);
char a_o_0(char anio[]);
//*************************************************************************
*************************************************************************
**********//
int main()
{
```

```
    srand(time(NULL));
    menu();
    return 0;
}
//*********************************************************************************
**********//
int msgs()
{
    int opc;
    system("cls");
    printf("------Menu------\n");
    printf("1.- Generar curp\n");
    printf("0.- Salir\n");
    opc = vali_int(0, 1, "Seleccione una opcion: ", "Opcion invalida");
    return opc;
}
//*********************************************************************************
**********//
void menu()
{
    int opc;
    do{
        opc = msgs();
        switch(opc)
        {
            case 1:
                curp();
                break;
        }
    }while (opc != 0);
}
//*********************************************************************************
**********//
void curp()
{
    char curp[19];
    char ap_pat[20], ap_mat[20], nombre1[20], nombre2[20], nombres3[40], anio[5],
mes[3], dia[3], genero[2], ent_fed[3];
    int dias_mes, i, correct;

    do{
        system("cls");
```

```
        vali_cad_obg(ap_pat, "Ingrese su apellido paterno: ");
        system("cls");
        vali_cad(ap_mat, "(Si solo tiene un apellido deje en blanco y presione
'Enter') \nIngrese su apellido materno: ");
        system("cls");
        vali_cad_obg(nombre1, "Ingrese su primer nombre: ");
        system("cls");
        vali_cad(nombre2, "(Si no tiene segundo nombre deje en blanco y presione
'Enter') \nIngrese su segundo nombre: ");
        system("cls");
        if(nombre2[0] != '\0')
        {
            vali_cad(nombres3, "(Si no tiene mas nombres deje en blanco y
presione 'Enter') \nSi tiene mas nombres ingreselos: ");
            system("cls");
        }
        vali_num_cad(anio, 1900, 2023, 4, "Ingrese su anio de nacimiento: ",
"Anio invalido");
        system("cls");
        vali_num_cad(mes, 1, 12, 2, "Ingrese su mes de nacimiento con numeros (1-
12): ", "Mes invalido");
        system("cls");
        dias_mes = meses(anio, mes);
        vali_num_cad(dia, 1, dias_mes, 2, "Ingrese su dia de nacimiento: ", "Dia
invalido");
        system("cls");
        vali_num_cad(genero, 1, 2, 1, "1.- Hombre \n2.- Mujer \nSeleccione una
opcion: ", "Opcion invalida");
        system("cls");
        for(i=0;i<33;i++)
        {
            printf("%d.- %s\n",i+1, ent[i]);
        }
        vali_num_cad(ent_fed, 1, 33, 2, "Ingrese su entidad federativa: ",
"Entidad invalida");
        system("cls");

        uno_a_dos_digitos(mes);
        uno_a_dos_digitos(dia);

        enie_mayus(ap_pat);
        enie_mayus(ap_mat);
        enie_mayus(nombre1);
        enie_mayus(nombre2);
        enie_mayus(nombres3);
```

```
        printf("Primer nombre: %s\n", nombre1);
        if(nombre2[0] != '\0')
        {
            printf("Segundo nombre: %s\n", nombre2);
        }
        else
        {
            printf("Segundo nombre: N/A\n");
        }
        if(nombres3[0] != '\0')
        {
            printf("Otros nombres: %s\n", nombres3);
        }
        else
        {
            printf("Otros nombres: N/A\n");
        }
        printf("Apellido Paterno: %s\n", ap_pat);
        if(ap_mat[0] != '\0')
        {
            printf("Apellido materno: %s\n", ap_mat);
        }
        else
        {
            printf("Apellido materno: N/A\n");
        }
        printf("Fecha de nacimiento: %s/%s/%s\n", dia, mes, anio);
        printf("Entidad federativa: %s\n\n", ent[atoi(ent_fed)-1]);
        correct = vali_int(0, 1, "Estos datos son correctos? \n1.- Si \n0.- No
\nSeleccione una opcion: ", "Valor Invalido");
        system("cls");
    }while(correct == 0);

    enie(ap_pat);
    enie(ap_mat);
    enie(nombre1);
    enie(nombre2);

    if(dela_del(ap_pat) == 0)
    {
        curp[0] = ap_pat[0];
        curp[1] = busq_vocal(ap_pat);
    }
    else
```

```
    {
        if(dela_del(ap_pat) == 1)
        {
            curp[0] = ap_pat[4];
            curp[1] = busq_vocal_del(ap_pat);
        }
        else
        {
            if(dela_del(ap_pat) == 2)
            {
                curp[0] = ap_pat[6];
                curp[1] = busq_vocal_dela(ap_pat);
            }
        }
    }
    if(ap_mat[0] != '\0')
    {
        if(dela_del(ap_mat) == 0)
        {
        curp[2] = ap_mat[0];
        }
        else
        {
            if(dela_del(ap_mat) == 1)
            {
                curp[2] = ap_mat[4];
            }
            else
            {
                if(dela_del(ap_mat) == 2)
                {
                    curp[2] = ap_mat[6];
                }
            }
        }
    }
    else
    {
        curp[2] = 'X';
    }
    if(jose_maria(nombre1, nombre2) == 1)
    {
        curp[3] = nombre1[0];
    }
    else
```

```
    {
        if(jose_maria(nombre1, nombre2) == 0)
        {
            curp[3] = nombre2[0];
        }
    }
    curp[4] = '\0';
    groserias(curp);
    curp[4] = anio[2];
    curp[5] = anio[3];
    curp[6] = mes[0];
    curp[7] = mes[1];
    curp[8] = dia[0];
    curp[9] = dia[1];
    curp[10] = gen(genero);
    entidad(ent_fed, curp);/*Aqui se alteran los campos 11 y 12 del curp*/
    if(dela_del(ap_pat) == 0)
    {
        curp[13] = busq_cons(ap_pat);
    }
    else
    {
        if(dela_del(ap_pat) == 1)
        {
            curp[13] = busq_cons_del(ap_pat);
        }
        else
        {
            if(dela_del(ap_pat) == 2)
            {
                curp[13] = busq_cons_dela(ap_pat);
            }
        }
    }
    if(ap_mat[0] != '\0')
    {
        if(dela_del(ap_mat) == 0)
        {
        curp[14] = busq_cons(ap_mat);
        }
        else
        {
            if(dela_del(ap_mat) == 1)
            {
                curp[14] = busq_cons_del(ap_mat);
```

```
            }
            else
            {
                if(dela_del(ap_mat) == 2)
                {
                    curp[14] = busq_cons_dela(ap_mat);
                }
            }
        }
    }
    else
    {
        curp[14] = 'X';
    }
    if(jose_maria(nombre1, nombre2) == 1)
    {
        curp[15] = busq_cons(nombre1);
    }
    else
    {
        if(jose_maria(nombre1, nombre2) == 0)
        {
            curp[15] = busq_cons(nombre2);
        }
    }
    curp[16] = a_o_0(anio);
    curp[17] = ((rand() % 9) + 1) + '0';
    curp[18] = '\0';

    printf("Su curp es: ");
    for (i = 0; i < 18; i++)
    {
        printf("%c", curp[i]);
    }
    printf("\n");
    system("pause");
}
//*****************************************************************************
*****************************************************************************
**********//
int meses(char anio[], char mes[])
{
    int meses[12], xanio, xmes;
    xanio = atoi(anio);
    xmes = atoi(mes);
```

```
    meses[0] = 31;/*Enero*/
    if(xanio % 4 == 0 && xanio % 100 != 0 || xanio % 400 == 0)
    {
        meses[1] = 29;/*Febrero*/
    }
    else
    {
        meses[1] = 28;/*Febrero*/
    }
    meses[2] = 31;/*Marzo*/
    meses[3] = 30;/*Abril*/
    meses[4] = 31;/*Mayo*/
    meses[5] = 30;/*Junio*/
    meses[6] = 31;/*Julio*/
    meses[7] = 31;/*Agosto*/
    meses[8] = 30;/*Septiembre*/
    meses[9] = 31;/*Octubre*/
    meses[10] = 31;/*Noviembre*/
    meses[11] = 31;/*Diciembre*/

    return meses[xmes-1];
}
//*****************************************************************************
*****************************************************************************
**********//
void groserias(char cadena[])
{
    int i;

    for(i=0; i<(sizeof(palabras_groseras)/sizeof(palabras_groseras[0])); i++)
    {
        if(strcmp(cadena, palabras_groseras[i]) == 0)
        {
            cadena[1] = 'X';
        }
    }
}
//*****************************************************************************
*****************************************************************************
**********//
char gen(char cadena[])
{
    int xcadena;
```

```
    xcadena = atoi(cadena);
    if(xcadena == 1)
    {
        return 'H';
    }
    else
    {
        return 'M';
    }
}
//*****************************************************************************
*****************************************************************************
**********//
void entidad(char cadena[], char curp[])
{
    int xcadena;

    xcadena = atoi(cadena);
    xcadena--;

    curp[11] = ent_abrev[xcadena][0];
    curp[12] = ent_abrev[xcadena][1];
}
//*****************************************************************************
*****************************************************************************
**********//
int jose_maria(char nombre1[], char nombre2[])
{
    if(strcmp(nombre1, "JOSE") == 0 && nombre2[0] != '\0' )
    {
        return 0;
    }
    else
    {
        if(strcmp(nombre1, "MARIA") == 0 && nombre2[0] != '\0')
        {
            return 0;
        }
    }


        return 1;
}
```

```
//*****************************************************************************
*****************************************************************************
**********//
char a_o_0(char anio[])
{
    int xanio;

    xanio = atoi(anio);
    if(xanio < 2000)
    {
        return '0';
    }
    else
    {
        if(xanio < 2010)
        {
            return 'A';
        }
        else
        {
            if(xanio < 2020)
            {
                return 'B';
            }
            else
            {
                if(xanio < 2030)
                {
                    return 'C';
                }
            }
        }
    }
}
```

LIBRERÍA

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define TRUE 1
#define FALSE 0
#define NUM_COL 4

int vali_int(int ran_inf, int ran_sup, char *msg, char *msg_error)
{
    int num;
    char char_num[25];

    do{
        printf("%s", msg);
        fflush(stdin);
        gets(char_num);
        num = atoi(char_num);

        if (num < ran_inf || num > ran_sup)
        {
            printf("%s\n", msg_error);
        }
    }while(num < ran_inf || num > ran_sup);

    return num;
}

long vali_long(long ran_inf, long ran_sup, char *msg, char *msg_error)
{
    long num;
    char char_num[25];

    do{
        printf("%s", msg);
        fflush(stdin);
        gets(char_num);
        num = atoi(char_num);

        if (num < ran_inf || num > ran_sup)
        {
            printf("%s\n", msg_error);
        }
```

```
    }while(num < ran_inf || num > ran_sup);

    return num;
}

void vali_num_cad(char cadena[], int ran_inf, int ran_sup, int max, char *msg,
char *msg_error)
{
    char xnum[max];
    int num;

    do{
        printf("%s", msg);
        fflush(stdin);
        gets(xnum);
        num = atoi(xnum);
        if((num < ran_inf || num > ran_sup) || strlen(xnum) > max)
        {
            printf("%s\n", msg_error);
        }
    }while((num < ran_inf || num > ran_sup) || strlen(xnum) > max);

    strcpy(cadena, xnum);

}

int busq_sec(int vect[], int len, int val)
{
    int i;
    for(i=0; i<len; i++)
    {
        if(val == vect[i])
        {
            return i;
        }
    }
    return -1;
}

char busq_vocal(char cadena[])
{
    int tam, i;
    tam = strlen(cadena);

    for(i=1; i<tam; i++)
```

```
    {
        if(cadena[i]=='A' || cadena[i]=='E' || cadena[i]=='I' || cadena[i]=='O'
|| cadena[i]=='U')
        {
            return cadena[i];
        }
    }

    return 'X';
}

char busq_vocal_dela(char cadena[])
{
    int tam, i;
    tam = strlen(cadena);

    for(i=7; i<tam; i++)
    {
        if(cadena[i]=='A' || cadena[i]=='E' || cadena[i]=='I' || cadena[i]=='O'
|| cadena[i]=='U')
        {
            return cadena[i];
        }
    }

    return 'X';
}

char busq_vocal_del(char cadena[])
{
    int tam, i;
    tam = strlen(cadena);

    for(i=5; i<tam; i++)
    {
        if(cadena[i]=='A' || cadena[i]=='E' || cadena[i]=='I' || cadena[i]=='O'
|| cadena[i]=='U')
        {
            return cadena[i];
        }
    }

    return 'X';
}
```

```
char busq_cons(char cadena[])
{
    int tam, i;
    tam = strlen(cadena);

    for(i=1; i<tam; i++)
    {
        if(cadena[i]!='A' && cadena[i]!='E' && cadena[i]!='I' && cadena[i]!='O'
&& cadena[i]!='U')
        {
            return cadena[i];
        }
    }

    return 'X';
}

char busq_cons_dela(char cadena[])
{
    int tam, i;
    tam = strlen(cadena);

    for(i=7; i<tam; i++)
    {
        if(cadena[i]!='A' && cadena[i]!='E' && cadena[i]!='I' && cadena[i]!='O'
&& cadena[i]!='U')
        {
            return cadena[i];
        }
    }

    return 'X';
}

char busq_cons_del(char cadena[])
{
    int tam, i;
    tam = strlen(cadena);

    for(i=5; i<tam; i++)
    {
        if(cadena[i]!='A' && cadena[i]!='E' && cadena[i]!='I' && cadena[i]!='O'
&& cadena[i]!='U')
        {
            return cadena[i];
```

```
        }
    }

    return 'X';
}

int existe_vect(int num, int *vect, int len)
{
    int i;
    for(i=0; i <= len; i++)
    {
        if (num == vect[i])
        {
            return 1;
        }
    }
    return 0;
}

int existe_mat(int num, int mat[][NUM_COL], int len)
{
    int i, j;
    for(i=0; i <= len; i++)
    {
        for(j=0; j <= len; j++)
        {
            if (num == mat[i][j])
            {
                return 1;
            }
        }
    }
    return 0;
}

void uno_a_dos_digitos(char cadena[2])
{
    int xnum_temp;
    xnum_temp = atoi(cadena);

    if(xnum_temp < 10)
    {
        cadena[0] = '0';
        cadena[1] = xnum_temp + '0';
        cadena[2] = '\0';
```

```
    }
}

int comp_esp(char cadena[])
{
    int tam;
    int i;

    tam = strlen(cadena);
    if(cadena[0] == '\0')
            return 0;
    for(i = 1; i <= tam; i++)
    {
        if(cadena[i] == ' ' && cadena[i+1] == ' ')
            return -1;
    }
}

int solo_letras(char cadena[])
{
    int tam;
    int i;

    tam = strlen(cadena);
    if (comp_esp(cadena) == 0)
    {
        return 0;
    }
    for(i =  0; i <= tam; i++)
    {
        if(cadena[i] < 'A' || cadena[i] > 'z')
        {
            return -1;
        }
        if(cadena[i] > 'Z' && cadena[i] < 'a')
        {
            return -1;
        }

        return 0;
    }
}

void vali_cad(char cadena[], char *msg)
{
```

```
    int tam, band;

    band = 1;
    do{

        printf("%s", msg);
        fflush(stdin);
        gets(cadena);

        if (solo_letras(cadena) == 0)
        {
            if (comp_esp(cadena) == -1)
            {
                printf("No se permite doble espacio\n");
                continue;
            }
        }
        else
        {
            if(cadena[0] == ' ')
            {
                printf("No se permiten espacios\n");
                continue;
            }
            else
            {
                printf("Solo puedes ingresar letras\n");
                continue;
            }
        }

        strupr(cadena);
        band = 0;

    }while(band == 1);
}

int comp_esp_obg(char cadena[])
{
    int tam;
    int i;

    tam = strlen(cadena);
    for(i = 1; i <= tam; i++)
    {
```

```
        if(cadena[i] == ' ' && cadena[i+1] == ' ')
            return -1;
    }
}

int solo_letras_obg(char cadena[])
{
    int tam;
    int i;

    tam = strlen(cadena);
    for(i =  0; i <= tam; i++)
    {
        if(cadena[i] < 'A' || cadena[i] > 'z')
        {
            return -1;
        }
        if(cadena[i] > 'Z' && cadena[i] < 'a')
        {
            return -1;
        }

        return 0;
    }
}

void vali_cad_obg(char cadena[], char *msg)
{
    int tam, band;

    band = 1;
    do{

        printf("%s", msg);
        fflush(stdin);
        gets(cadena);

        if (solo_letras_obg(cadena) == 0)
        {
            if (comp_esp_obg(cadena) == -1)
            {
                printf("No se permite doble espacio\n");
                continue;
            }
        }
```

```
        else
        {
            if(cadena[0] == ' ')
            {
                printf("No se permiten espacios\n");
                continue;
            }
            else
            {
                printf("Solo puedes ingresar letras\n");
                continue;
            }
        }

        strupr(cadena);
        band = 0;

    }while(band == 1);
}

void enie(char cadena[])
{
    int i, len;

    len = strlen(cadena);

    for(i=0; i<len; i++)
    {
        if (cadena[i] == -92 || cadena[i] == -91)
        {
            cadena[i] = 'X';
        }
    }
}

void enie_mayus(char cadena[])
{
    int i, len;

    len = strlen(cadena);

    for(i=0; i<len; i++)
    {
        if (cadena[i] == -92)
        {
```

```
            cadena[i] = -91;
        }
    }
}

int dela_del(char ap[])
{
    if(ap[0] == 'D' && ap[1] == 'E' && ap[2] == ' ' && ap[3] == 'L' && ap[4] ==
'A' && ap[5] == ' ')
    {
        return 2;
    }
    else
    {
        if(ap[0] == 'D' && ap[1] == 'E' && ap[2] == 'L' && ap[3] == ' ')
        {
            return 1;
        }
    }

    return 0;


}
```