ALGORITMOS (Alternativas múltiples)

Utilizando las estructuras de alternativas, desarrolle el algoritmo correspondiente para cada problema que se le presenta.

Recuerde seguir las reglas para la escritura de algoritmo, reglas y recomendaciones para darle nombre adecuado a las áreas de memoria, indentación y formato establecido para las estructuras.

1. Se solicita diseñar el algoritmo que:
   1. Pida por teclado el número (dato entero) de un día de la semana.
   2. Muestre por pantalla el nombre (dato cadena) correspondiente a dicho día.

Nota: Si el número de día introducido es menor que 1 ó mayor que 7, se mostrará el mensaje: "ERROR: Día incorrecto.".

Algoritmo DiaDeSemana

{

String diaSemana;

Entero numSemana;

Leer(numSemana);

switch (numSemana)

{

1: {

diaSemana = ‘Domingo’;

break;

}

2: {

diaSemana = ‘Lunes’;

break;

}

3: {

diaSemana = ‘Martes’;

break;

}

4: {

diaSemana = ‘Miércoles’;

break;

}

5: {

diaSemana = ‘Jueves’;

break;

}

6: {

diaSemana = ‘Viernes’;

break;

}

7: {

diaSemana = ‘Sabado’;

break;

}

Sino

{

Printf(“Dia incorrecto”);

break;

}

}

Printf(diaSemana);

}

1. En la siguiente tabla se muestran las categorías a las que pertenecen los signos del zodiaco:

|  |  |
| --- | --- |
| Signo | Categoría |
| 1. Aries | Fuego |
| 2. Tauro | Tierra |
| 3. Géminis | Aire |
| 4. Cáncer | Agua |
| 5. Leo | Fuego |
| 6. Virgo | Tierra |
| 7. Libra | Aire |
| 8. Escorpio | Agua |
| 9. Sagitario | Fuego |
| 10. Capricornio | Tierra |
| 11. Acuario | Aire |
| 12. Piscis | Agua |

Se solicita diseñar un algoritmo que:

* 1. Muestre el listado de los signos del zodíaco, con sus números asociados.
  2. Pida por teclado un número (dato entero) asociado a un signo del zodíaco.
  3. Muestre la categoría a la que pertenece el signo del zodíaco seleccionado.

Nota: Si el número introducido por el usuario, no está asociado a ningún signo del zodíaco, se mostrará el mensaje: "ERROR: <número> no está asociado a ningún signo.".

Algoritmo SignosDelSodicao

{

String signo, categoria;

Entero num;

Leer(num);

Printf()

switch (num)

{

1: {

Signo = ‘aries’;

Categoría = ‘fuego’;

Break;

}

2:{

Signo = ‘tauro’;

Categoria = ‘tierra’;

Break;

}

3: {

Signo = ‘geminis’;

Categoría = ‘aire’;

Break;

}

4: {

Signo = ‘cancer’;

Categoría = ‘agua’;

Break;

}

5: {

Signo = ‘leo’;

Categoría = ‘fuego’;

Break;

}

6: {

Signo = ‘virgo’;

Categoria = ‘tierra’;

Break;

}

7: {

Signo = ‘libra’;

Categoria = ‘aire’;

Break;

}

8: {

Signo = ‘escorpio’;

Categoria = ‘agua’;

Break;

}

9: {

Signo = ‘sagitario’;

Categoria = ‘fuego’;

Break;

}

10: {

Signo = ‘capricornio’;

Categoria = ‘tierra’;

Break;

}

11: {

Signo = ‘acuario’;

Categoria = ‘aire’;

Break;

}

12: {

Signo = ‘piscis’;

Categoria = ‘agua’;

Break;

}

}

Printf(“Signo:” signo, “Categoria:” categoria);

}

1. Se solicita diseñar el algoritmo que:
   1. Muestre un listado con los nombres de los meses, enumerados desde 1 hasta 12
   2. Luego pida por teclado el número (dato entero) de un día del mes.
   3. Muestre por pantalla cuántos días tiene el mes seleccionado.

Nota: Si el número de mes introducido es menor que 1 ó mayor que 12, se mostrará el mensaje: "ERROR: Mes incorrecto.".

Algoritmo Meses

{

String mes;

Entero num, diasMes;

Leer(num);

Switch (num)

{

1: {

Mes = ‘enero’;

diasMes = 31;

break;

}

2: {

Mes = ‘febrero’;

diasMes = 28;

break;

}

3: {

Mes = ‘marzo’;

diasMes = 31;

break;

}

4: {

Mes = ‘abril’;

diasMes = 30;

break;

}

5: {

Mes = ‘mayo’;

diasMes = 31;

break;

}

6: {

Mes = ‘junio’;

diasMes = 30;

break;

}

7: {

Mes = ‘julio’;

diasMes = 31;

break;

}

8: {

Mes = ‘agosto’;

diasMes = 31;

break;

}

9: {

Mes = ‘septimebre’;

diasMes = 30;

break;

}

10: {

Mes = ‘octubre’;

diasMes = 31;

break;

}

11: {

Mes = ‘noviembre’;

diasMes = 30;

break;

}

12: {

Mes = ‘diciembre’;

diasMes = 31;

break;

}

}

Printf(mes + “, tiene, ” + diasMes);

}

1. Según políticas de la empresa, de acuerdo al estado civil de sus colaboradores, se solicita una colaboración para las fiestas patronales.

|  |  |  |
| --- | --- | --- |
| **Estado Civil** | **Código** | **Colaboración** |
| Soltero | ‘S’ | 25.00 |
| Casado | ‘C’ | 20.00 |
| Divorciado | ‘D’ | 15.00 |
| Unido | ‘U’ | 10.00 |
| Viudo | ‘V’ | 5.00 |

Se solicita leer el nombre del colaborador y el código del estado civil. De acuerdo a lo que corresponda, desplegar el mensaje: SE LE COMUNICA A **xxxxxxxxxx** QUE SU COLABORACION PARA LAS FIESTAS PATRONALES ES DE: **99.99**. GRACIAS POR SU ATENCION.

Algoritmo FiestaPatronal

{

String nombre, codigo;

Real colaboracion;

Leer(nombre, codigo);

Switch (codigo)

{

Case ‘S’: {

colaboración = 25.00;

break;

}

Case ‘C’: {

colaboración = 20.00;

break;

}

Case ‘D’: {

colaboración = 15.00;

break;

}

Case ‘U’: {

colaboración = 10.00;

break;

}

Case ‘V’: {

colaboración = 5.00;

break;

}

}

Printf(“Se le comunica a ” + nombre “, que su colaboración para las fiestas patronales es de” + colaboracion);

}

1. La Universidad Tecnológica de Panamá cuenta con 6 facultades a nivel nacional. Se solicita preparar un menú que le permita acceder a una facultad en especial. Una vez que elija la Facultad, debe enviar un mensaje de bienvenida.

UNIVERSIDAD TECNOLOGICA DE PANAMA

LISTADO DE FACULTADES

1. INGENIERIA CIVIL
2. INGENIERIA ELECTRICA
3. INGENIERIA INDUSTRIAL
4. INGENIERIA MECANICA
5. INGENIERIA DE SISTEMAS COMPUTACIONALES
6. CIENCIAS Y TECNOLOGIA

SELECCIONE OPCION: \_\_\_

BIENVENIDOS A LA FACULTAD DE xxxxxxxxxxxxx

Algoritmo Utp

{

String facultad;

Entero numero;

Leer(numero);

Switch (numero)

{

1 : {

facultad = “INGENIERIA CIVIL”;

}

2 : {

facultad = “INGENIERIA ELECTRICA”;

}

3 : {

facultad = “INGENIERIA INDUSTRIAL”;

}

4 : {

facultad = “INGENIERIA MECANICA”;

}

5 : {

facultad = “INGENIERIA SISTEMAS COMPUTACIONALES”;

}

6 : {

facultad = “CIENCIAS Y TECNOLOGIAS”;

}

}

Printf(“BIENVENIDOS A LA FACULTAD DE” + facultad);

}

1. Construya un algoritmo que permita calcular lo que hay que pagarle a un trabajador teniendo en cuenta su sueldo y las horas extras trabajadas. Para el pago de horas extras se toma en cuenta la categoría del trabajador:

|  |  |
| --- | --- |
| Categoría | Precio de hora extra |
| 1 | 3.65 |
| 2 | 4.75 |
| 3 | 5.65 |
| 4 | 7.50 |

Se debe leer: salario del trabajador, cantidad de horas extras, categoría.

Imprimir el salario que se le tiene que pagar al trabajador.

Algoritmo PagoTrabajador

{

String nombre;

Float salarioBase, hrsXtra, pagoXtra, salarioFinal,;

Int categoría;

Leer(nombre, salarioBase, hrsXtra, categoria);

Switch (categoria)

{

1 : {

pagoXtra = 3.65;

}

2 : {

pagoXtra = 4.75;

}

3 : {

pagoXtra = 5.65;

}

4 : {

pagoXtra = 7.50;

}

}

hrsXtra = hrsXtra \* pagoXtra

salarioFinal = salarioBase + hrsXtra;

printf(“Nombre:” nombre, “Salario:” salarioFinal);

}

1. Construya un algoritmo, que al recibir como datos el nivel académico de un profesor de una Universidad, así como su salario, incremente éste último siguiendo las especificaciones de la tabla e imprima tanto el nivel del profesor como su nuevo salario.

|  |  |
| --- | --- |
| Nivel | Incremento |
| Profesor adjunto | 4.2% |
| Profesor auxiliar | 5.6% |
| Profesor asistente | 6.8% |
| Profesor titular | 7.4% |

Se debe leer el nivel académico del profesor y el salario

Imprimir el nivel del profesor y el nuevo salario.

Algoritmo Profesor

{

String nivelAc;

Float salario, incremento, nuevoSalario, porcentage;

Leer(nivelAc, salario);

Switch (nivelAc)

{

Case “Profesor adjunto” : {

pocentage = 0.042

}

Case “Profesor auxiliar” : {

porcentage = 0.056

}

Case “Profesor asistente” : {

porcentage = 0.068

}

Case “Profesor titular” : {

porcentage = 0.074

}

}

incremento = salario \* porcentaje;

nuevoSalario = salario + incremento;

printf(“Nivel de Profesor: ” nivelAc, “Nuevo salario: ” nuevoSalario);

}