9.- Operaciones con fechas.

9.1.- Lectura de fechas.

Las fechas son un tipo de datos que nos sirven para medir el tiempo a gran escala, el tiempo se utiliza en los negocios para plazos comerciales, vencimientos de documentos, validez y vigencia de derechos y licencias y un sin número de conceptos útiles y muy comunes. Debido a lo anterior las operaciones con fechas son por demás relevantes en cualquier lenguaje de programación. El inovaProg proporciona facilidades que permiten la lectura, escritura y operaciones aritméticas básicas con fechas.

La operación de lectura de una fecha se lleva a cabo de la siguiente manera:

```
String fechaFacturacion="";
fechaFacturacion =Ip.wleeFecha(
        "Escriba la fecha inicial (dd/mm/aaaa) ",
        fechaFacturacion);
```

Aunque la variable esta declarada como cadena de caracteres, el inovaProg la valida para que cumpla con los requisitos del formato dd/mm/aaaa y que sea una fecha válida, de esta forma puede utilizarse en otras operaciones posteriormente.
La sintaxis general queda como siguen:

<variableCadenaParaFecha>=Ip.wleeFecha(
<varableConMensaje>,
<variableCadenaParaFecha>;

9.2.- Aumento de días a una fecha para obtener otra.

La primera operación común con fechas es la suma de un número determinado de días a una fecha dada para obtener una nueva fecha.

Para hacer esto contamos con las siguientes instrucciones:
Suponiendo que hemos leído la fechaFacturacion en una cadena de caracteres como se muestra arriba y se le desea aumentar 15 días a dicha fecha para obtener la fechaVencimiento podemos utilizar la instrucciones siguientes:

```
String fechaVencimiento="";
fechaVencimiento =
Ip.aumentaDias(15,fechaFacturacion);
```

La fechaVencimiento es una fecha posterior en 15 días a la fecha contendida en fechaFacturacion. La sintaxis general queda como se muestra:

<variableCadenaParaFechaPosterior>= Ip.aumentaDias (
<varableEnteraConNumeroDeDias>,
<variableCadenaParaFechaOriginal>;

Una fecha puede almacenarse en una variable de cadena o en una variable tipo Date para convertir una cadena a una fecha tipo Date se cuenta con la siguiente facilidad: convCadenaAFecha(String s) que se utiliza de la siguiente manera:

```
Date fechaTipoDateFac = Ip.convCadenaAFecha(
                         fechaFacturacion);
```

Asumiendo que fechaFacturacion contiene una cadena de caracteres que representa una fecha válida.
La sintaxis general se muestra a continuación:

<variableTipoDate>= Ip.convCadenaAFecha (
<variableCadenaConFecha>;

A una variable tipo Date también le podemos sumar un número determinado de días:

```
Date fechaTipoDateVenc = null;
Date fechaTipoDateFac = null;
String fechaFacturacion="";
fechaFacturacion =Ip.wleeFecha(
        "Escriba la fecha inicial (dd/mm/aaaa) ",
        fechaFacturacion);
fechaTipoDateFac = Ip.convCadenaAFecha(
                   fechaFacturacion);
fechaTipoDateVenc = Ip.aumentaDias(15,
                   fechaTipoDateFac);
```

La sintaxis para sumar un determinado número de días a una fecha tipo Date queda como:

<variableTipoDateParaFechaPosterior>= Ip.aumentaDias (
<varableEnteraConNumeroDeDias>,
<variableTipoDateParaFechaOriginal>;

Una manera de obtener la fecha del día actual es inicializar una fecha tipo Date de la siguiente manera:

```
Date fechaDeHoy = new Date();
```

9.3.- Disminución de días a una fecha para obtener otra.

La otra operación común es la resta de días o una fecha para obtener una fecha anterior.

Para hacer esto contamos con las siguientes instrucciones:
Suponiendo que hemos leído la fechaFinal en una cadena de caracteres y se le desea disminuir 15 días a dicha fecha para obtener la fechaInicial podemos utilizar la instrucciones siguientes:

```
String fechaInicial="";
fechaInicial = Ip.disminuyeDias(15,fechaFinal);
```

La fechaInicial es una fecha anterior en 15 días a la fecha contendida en fechaFinal. La sintaxis general queda como se muestra:

<variableCadenaParaFechaAnterior>= Ip.disminuyeDias (
<varableEnteraConNumeroDeDias>,
<variableCadenaParaFechaOriginal>;

A una variable tipo Date también le podemos restar un número determinado de días:

```
Date fechaTipoDateIni = null;
Date fechaTipoDateFin = null;
String fechaFinal="";
fechaFacturacion =Ip.wleeFecha(
        "Escriba la fecha final (dd/mm/aaaa) ",
        fechaFinal);
fechaTipoDateFin = Ip.convCadenaAFecha(
                   fechaFinal);
fechaTipoDateIni = Ip.disminuyeDias(15,
                   fechaTipoDateFin);
```

La sintaxis para sumar un determinado número de días a una fecha tipo Date queda como:

<variableTipoDateParaFechaAnterior>= Ip.disminuyeDias (
<varableEnteraConNumeroDeDias>,
<variableTipoDateParaFechaOriginal>;

9.4.- Calculo del número de días entre dos fechas.

Es posible calcular el número de días que existen entre dos fechas. Por ejemplo el siguiente programa calcula los días vividos por el usuario.

```
int diasVividos = 0;
String fechaNacimiento="";
String fechaDeHoy= convFechaACadena(new Date());
fechaNacimiento =Ip.wleeFecha(
"Escriba su fecha de nacimiento (dd/mm/aaaa) ",
        fechaNacimiento);
diasVividos = Ip.diferenciaFechas
              (fechaNacimiento, fechaDeHoy));
Ip.escribemensaje("Usted ha vivido +
                   diasVividos + " dias ");
```

La sintaxis general para la diferencia de fechas es:

<variableEnteraParaNumDias>= Ip. diferenciaFechas (
<varableCadenaConFechaAnterior>,
<variableCadenaConFechaPosterior>;

O bien:

<variableEnteraParaNumDias>= Ip. diferenciaFechas (
<varableTipoDateConFechaAnterior>,
<variableTipoDateConFechaPosterior>;

9.5.- Formatos en los que se puede imprimir una fecha.

Existen dos maneras de obtener la fecha actual:

1. Utilizando el programa (clase) Date predefinido por Java como ya se había discutido anteriormente. Por ejemplo:
   ```
   Date fechaActual = new Date();
   ```

2. Utilizando el programa (clase) GregorianCalendar por ejemplo:
   ```
   GregorianCalendar calendario =
                  new GregorianCalendar();
   ```


Para darle forma a la fecha se cuenta con los siguientes programas (clases predefinas por Java):
DateFormat se utiliza para usar formatos predefinidos.
SimpleDateFormat se utiliza para definir formatos arbitrarios por parte del usuario.

Para utilizar el Date format podemos observar el sigeuinte ejemplo:

```
//Definir la variables
Date fechaActual = new Date();
 //Definir la instancia de los formatos predefinidos
DateFormat formPorDefecto = DateFormat.getDateInstance();
DateFormat formCorto = DateFormat.getDateInstance(DateFormat.SHORT);
DateFormat formMediano = DateFormat.getDateInstance(DateFormat.MEDIUM);
DateFormat formLargo = DateFormat.getDateInstance(DateFormat.LONG);
DateFormat formCompleto = DateFormat.getDateInstance(DateFormat.FULL);
 //Utilizar en la variable previamente definida tipo Date los formatos
String fechaPorDefecto = formPorDefecto.format(fechaActual);
String fechaCorta      = formCorto.format(fechaActual);
String fechaMediana    = formMediano.format(fechaActual);
String fechaLarga      = formLargo.format(fechaActual);
String fechaCompleta   = formCompleto.format(fechaActual);
```

Y para utilizar el SimpleDateFormat tenemos el siguiente código:

```
//Definir una instancia del calendario
GregorianCalendar calendario = new GregorianCalendar();
//Generar formatos definidos arbitrariamente por el usuario.
SimpleDateFormat formFechaYHora =
                 new SimpleDateFormat("dd-MMMM-yyyy HH:mm:ss");
SimpleDateFormat formFechaMex =
                 new SimpleDateFormat("dd/MM/yyyy");
SimpleDateFormat formatoFechaUsa = new SimpleDateFormat("yyyy/MM/dd");
//Utilizar los formatos, puede usarse la fecha actual o del calendario
String fechaYHora     = formFechaYHora.format(fechaActual);
String fechaDeHoyMex  = formFechaMex.format(calendario.getTime());
String fechaUsa       = formatoFechaUsa.format(calendario.getTime());
```

También es posible extraer la hora o partes de ella a partir de una instancia de una fecha.

```
//Generar formatos definidos arbitrariamente
//por el usuario para la hora.
SimpleDateFormat formHora         = new SimpleDateFormat("HH");
SimpleDateFormat formMins         = new SimpleDateFormat("mm");
SimpleDateFormat formHoraCompleta = new SimpleDateFormat("HH:mm:ss");
//Utilizar los formatos en el calendario
String Hora         = formHora.format(calendario.getTime());
String Minutos      = formMins.format(calendario.getTime());
String HoraCompleta = formHoraCompleta.format(calendario.getTime());
```

El ejemplo adjunto 902FormatosDeFechaYHora.java ilustra el uso de las instrucciones anteriores.

9.6.- Actividades
Compile y ejecute los ejemplos agregando variables y formatos.

9.7.- Ejercicio propuesto:
Escriba un programa que solicite la fecha de nacimiento de una persona y posteriormente imprima el tiempo vivido por dicha persona en días, luego en minutos y finalmente en segundos.