Exemples
Programació estructurada

El codi Java dels exemples s'escriu en el mètode main de la classe principal.

exemple 009

Escriu un programa que calcula i visualitza la data del diumenge de pasqua d'un any llegit de teclat. Els càlculs a realitzar són
A = any % 19
B = any % 4
C = any % 7
D = (19 * A + 24) % 30
E = (2 * B + 4 * C + 6 * D + 5) % 7
dies = 22 + D + E
Sent dies el nombre de dies del mes de març, en la visualització posa el nom del mes correcte "Març" o "Abril".

De la lectura de l'enunciat separa

- les dades
  - any
- les accions
  - **calcula i visualitza** la data del diumenge de pasqua
  - d'un any **llegit** de teclat
- les condicions
  - en la visualització posa el nom del mes correcte "Març" o "Abril"

Defineix la informació a manejar

- l'any, és un **nombre enter** (int), any
- el nombre de dies del mes de març, és un **nombre enter** (int), dies
- càlculs intermedis, són **nombres enters** (int), A, B, C, D i E

Defineix les accions

Llig de teclat l'any, any

Realitza tots els càlculs intermedis indicats a l'enunciat.

Calcula el nombre de dies del mes de març, dies.

El mes de març té 31 dies, per tant, un valor superior a 31 índica que és una data d'abril i cal restar 31. Si dies > 31, llavors resta 31 a dies i posa "Abril" com a text, en cas contrari el valor de dies no canvia i posa "Març" com a text.

Visualitza un text amb el format "el diumenge de pasqua de any és el dies del mes" on s'expressa l'any, els dies i el literal del mes.

```
Scanner lector = new Scanner(System.in);
System.out.print("entra l'any ");
int any = lector.nextInt();
int A = any % 19;
int B = any % 4;
int C = any % 7;
int D = (19 * A + 24) % 30;
int E = (2 * B + 4 * C + 6 * D + 5) % 7;
int dies = 22 + D + E;
if (dies > 31) {        // és el mes d'Abril, la resta es realitza en la visualització
   System.out.println("el diumenge de pasqua de " + any + " és el " + (dies - 31) + " d'Abril");
} else {    // és el mes de Març
   System.out.println("el diumenge de pasqua de " + any + " és el " + dies + " de Març");
}
```

Sense usar referències intermèdies, una única expressió.

```
Scanner lector = new Scanner(System.in);
System.out.print("entra l'any ");
int any = lector.nextInt();
int dies = 22 + (19 * (any % 19) + 24) % 30 + (2 * (any % 4) + 4 * (any % 7) + 6 * ((19 * (any % 19) + 24) % 30) + 5)
% 7;
System.out.printf("el diumenge de pasqua de %d és el %d %s",
                  any, dies > 31 ? dies - 31 : dies, dies > 31 ? "d'Abril" : "de Març");
```

En el printf, la creació de la cadena de text final realitza dues vegades la pregunta dies > 31, mentre que en la versió anterior sols una.

Amb mètodes

getDiumengePasqua

El mètode permet obtindre el diumenge de pasqua.

private static String getDiumengePasqua(int any)

- rep l'any del qual calcula el diumenge de Pasqua, és un enter any
- torna una cadena de text
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Defineix la informació a manejar

- l'any, és un **nombre enter** (int), any és el paràmetre
- el nombre de dies del mes de març, és un **nombre enter** (int), dies
- càlculs intermedis, són **nombres enters** (int), A, B, C, D i E

Defineix les accions

Realitza tots els càlculs intermedis (A, B, C, D i E) indicats a l'enunciat.

Calcula el nombre de dies del mes de març, dies. El mes de març té 31 dies, per tant, un valor superior a 31 índica que és una data d'abril.

Si el valor de dies és major que 31, llavors resta 31 a dies i posa "Abril" com a text, en cas contrari el valor de dies no canvia i posa "Març" com a text.

Retorna una cadena de text amb el format "el diumenge de pasqua de any és el dies del mes" on s'expressa l'any, els dies i el literal del mes.

El codi del mètode és

```
private static String getDiumengePasqua(int any){
   int A = any % 19;
   int B = any % 4;
   int C = any % 7;
   int D = (19 * A + 24) % 30;
   int E = (2 * B + 4 * C + 6 * D + 5) % 7;
   int dies = 22 + D + E;
   if (dies > 31) {     // és el mes d'Abril, la resta es realitza en la visualització
      return "el diumenge de pasqua de " + any + " és el " + (dies - 31) + " d'Abril";
   }        // és el mes de Març
   return "el diumenge de pasqua de " + any + " és el " + dies + " de Març";
}
```

main

```
Scanner lector = new Scanner(System.in);
System.out.print("entra l'any ");
int any = lector.nextInt();
System.out.println(getDiumengePasqua(any));
```

exemple 006

Escriu un programa que augmenta el sou d'un empleat. Si cobra menys de 1000€ el sou puja un 8.5% i si és superior puja un 7.8%. visualitza el sou final.

De la lectura de l'enunciat separa

- les dades
  - el sou d'un empleat
  - el sou final
- les accions
  - **augmenta** el sou d'un empleat
  - **visualitza** el sou final
- les condicions
  - Si cobra menys de 1000€ el sou puja un 8.5%

- si és superior puja un 7.8%

Defineix la informació a manejar

- el sou i el sou final, és el mateix concepte en dos moments diferents, usa una única referència sou, són diners (euros), per tant, és un **nombre real** (double)

Defineix les accions

Llig de teclat el sou inicial del treballador, sou

Pel càlcul de l'augment crea la variable augment que contindrà euros, per tant, és real (double) i és un percentatge del sou inicial.

Si el sou inicial és menor o igual que 1000€ augment = sou * 8.5 / 100

Si el sou inicial és major que 1000€ augment = sou * 7.8 / 100

Sols es necessita un si amb un sinó, sols hi ha dos trams de sou a controlar.

El càlcul del sou final és sou = sou + augment

Visualitza el sou final.

```
Scanner lector = new Scanner(System.in);
lector.useLocale(Locale.UK);
System.out.print("sou: ");
double sou = lector.nextDouble();
double augment;
if (sou <= 1000) {
   augment = sou * 8.5 / 100;
} else {
   augment = sou * 7.8 / 100;
}
sou = sou + augment;
System.out.printf("sou final: %.2f€", sou);
```

Altra versió usant l'operador ?:, el resultat de (sou <= 1000 ? 8.5 : 7.8) és 8.5 (sou <= 1000 true) o 7.8 (sou <= 1000 false), el resultat s'assigna a augment.

```
Scanner lector = new Scanner(System.in);
lector.useLocale(Locale.UK);
System.out.print("sou: ");
double sou = lector.nextDouble();
double augment = (sou <= 1000 ? 8.5 : 7.8) * sou / 100;          // (sou <= 1000 ? 8.5 : 7.8) valdrà 8.5 o 7.8
sou = sou + augment;
System.out.printf("sou final: %.2f€", sou);
```

En el tractament següent es realitza sempre l'augment 7.8 i després es complementa per arribar a 8.5 si sou <= 1000, el complement és 0.7 que correspon a 8.5 – 7.8

```
Scanner lector = new Scanner(System.in);
lector.useLocale(Locale.UK);
System.out.print("sou: ");
double sou = lector.nextDouble();
double augment = sou * 7.8 / 100;
if (sou <= 1000) {
   augment = augment + sou * 0.7 / 100;
}
sou = sou + augment;
System.out.printf("sou final: %.2f€", sou);
```

exemple 010

Escriu un programa que llig de teclat un valor entre 0 i 9, i visualitza el nom del dígit en valencià ("zero", "un", "dos" …). Si s'introdueix un valor incorrecte, visualitza el missatge "no és un valor entre 0 i 9".

De la lectura de l'enunciat separa

- les dades
  - un valor entre 0 i 9
  - el nom del dígit
- les accions
  - llig de teclat un valor entre 0 i 9
  - visualitza el nom del dígit en valencià
  - visualitza el missatge "no és un valor entre 0 i 9"
- les condicions
  - e un valor entre 0 i 9
  - Si s'introdueix un valor incorrecte

Defineix la informació a manejar

- un valor entre 0 i 9, és **un nombre enter** (int), num
- el nom del dígit, és **una cadena de text** (literal)

Defineix les accions

Llig de teclat el valor del nombre, num.

Un enter pot prendre molts valors, però només hi ha dos tipus de respostes:

- num entre 0 i 9, mostra el text corresponent al valor introduït
- num no està entre 0 i 9, mostra el missatge d'error

És una selecció, pots usar l'expressió num < 0 || num > 9.

El bloc corresponent a has introduït un valor entre 0 i 9, necessita 10 comparacions (una per a cada dígit). Solució amb seleccions niades.

```
Scanner lector = new Scanner(System.in);
System.out.print("entra un valor entre 0 i 9 ");
int num = lector.nextInt();
if (num < 0 || num > 9) {                System.out.println("no és un valor entre 0 i 9");
} else {
  if (num > 8) {                         System.out.println("nou");
  } else {
    if (num > 7) {                       System.out.println("huit");
    } else {
      if (num > 6) {                     System.out.println("set");
      } else {
        if (num > 5) {                   System.out.println("sis");
        } else {
          if (num > 4) {                 System.out.println("cinc");
          } else {
            if (num > 3) {               System.out.println("quatre");
            } else {
              if (num > 2) {             System.out.println("tres");
              } else {
                if (num > 1) {           System.out.println("dos");
                } else {
                  if (num > 0) {         System.out.println("un");
                  } else {               System.out.println("zero");
                  }
                }
              }
            }
          }
        }
      }
    }
  }
}
```

Altra solució, els si no estan niats, hi ha 10 si.

```
if (num < 0 || num > 9) {
  System.out.println("no és un valor entre 0 i 9");
} else {
  if (num == 9) {      System.out.println("nou");   }
  if (num == 8) {      System.out.println("huit");   }
  if (num == 7) {      System.out.println("set");   }
  if (num == 6) {      System.out.println("sis");   }
  if (num == 5) {      System.out.println("cinc");   }
  if (num == 4) {      System.out.println("quatre");   }
  if (num == 3) {      System.out.println("tres");   }
  if (num == 2) {      System.out.println("dos");   }
  if (num == 1) {      System.out.println("un");   }
```

```
  if (num == 0) {      System.out.println("zero");   }
}
```

Altra solució, amb una selecció múltiple amb les opcions del 0 al 9 (case) i tots els altres valors (default) .

```
switch (num) {
   case 0:       System.out.println("zero");       break;
   case 1:      System.out.println("un");       break;
   case 2:       System.out.println("dos");       break;
   case 3:       System.out.println("tres");       break;
   case 4:       System.out.println("quatre");       break;
   case 5:       System.out.println("cinc");       break;
   case 6:       System.out.println("sis");       break;
   case 7:       System.out.println("set");       break;
   case 8:       System.out.println("huit");       break;
   case 9:       System.out.println("nou");       break;
   default:       System.out.println("no és un valor entre 0 i 9");
}
```

Altra solució, amb una selecció múltiple en notació lambda.

```
switch (num) {
   case 0 -> System.out.println("zero");
   case 1 -> System.out.println("un");
   case 2 -> System.out.println("dos");
   case 3 -> System.out.println("tres");
   case 4 -> System.out.println("quatre");
   case 5 -> System.out.println("cinc");
   case 6 -> System.out.println("sis");
   case 7 -> System.out.println("set");
   case 8 -> System.out.println("huit");
   case 9 -> System.out.println("nou");
   default -> System.out.println("no és un valor entre 0 i 9");
}
```

Transformant la notació lambda a una expressió.

```
System.out.println(
 switch (num) {
   case 0 -> "zero";
   case 1 -> "un";
   case 2 -> "dos";
   case 3 -> "tres";
   case 4 -> "quatre";
   case 5 -> "cinc";
   case 6 -> "sis";
   case 7 -> "set";
   case 8 -> "huit";
```

```
  case 9 -> "nou";
  default -> "no és un valor entre 0 i 9";
}};
```

Amb mètodes

getTextDigit

El mètode permet obtindre el text d'un dígit d'un enter.

private static String getTextDigit(int num)

- rep el nombre, que és un enter num, el paràmetre
- torna una cadena de text
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- un valor entre 0 i 9, és **un nombre enter** (int) amb la referència num
- el nom del dígit, és **una cadena de text**, txt

Defineix les accions

Un enter pot prendre molts valors, però només hi ha dos tipus de respostes:

- entre 0 i 9, assigna el nom del dígit corresponent al valor introduït al text
- no està entre 0 i 9, assigna el missatge d'error al text

Retorna la cadena de text txt.

El codi del mètode és

```
private static String getTextDigit(int num){
  String txt = "no és un valor entre 0 i 9";
  switch (num) {
    case 0 -> txt = "zero";
    case 1 -> txt = "un";
    case 2 -> txt = "dos";
    case 3 -> txt = "tres";
    case 4 -> txt = "quatre";
    case 5 -> txt = "cinc";
    case 6 -> txt = "sis";
    case 7 -> txt = "set";
    case 8 -> txt = "huit";
    case 9 -> txt = "nou";
  }
  return txt;
}
```

main

```
Scanner lector = new Scanner(System.in);
System.out.print("entra un valor entre 0 i 9 ");
int digit = lector.nextInt();
System.out.println(getTextDigit(digit));
```

exemple 011

Escriu un programa que calcula el descompte d'una compra en funció de l'import. Per a menys de 300€, no hi ha descompte, de 300€ a 600€ aplica un 5%, de 600€ a 1000€ aplica un 8%, de 1000€ a 3000€ aplica un 10%, de 3000€ a 6000€ aplica 13% i per damunt de 6000€ aplica un 20%. Visualitza l'import de la compra, el descompte i l'import final.

De la lectura de l'enunciat separa

- les dades
  - el descompte d'una compra
  - l'import
  - l'import final
- les accions
  - **calcula** el descompte
  - **visualitza** l'import de la compra, el descompte i l'import final
- les condicions
  - per a menys de 300€, no hi ha descompte
  - de 300€ a 600€ aplica un 5%
  - de 600€ a 1000€ aplica un 8%
  - de 1000€ a 3000€ aplica un 10%
  - de 3000€ a 6000€ aplica 13%
  - per damunt de 6000€ aplica un 20%

Defineix la informació a manejar

- l'import de la compra, és **un nombre real** (double), importCompra
- el descompte de la compra, és **un nombre real** (double), descompte
- l'import final de la compra, és **un nombre real** (double), importFinal

Defineix les accions

Llig de teclat l'import de la compra, importCompra

El percentatge de descompte a aplicar depén de l'import de la compra, per a menys de 300€ s'aplica un 0%, de 300€ a 600€ s'aplica un 5%, de 600€ a 1000€ s'aplica un 8%, de 1000€ a 3000€ s'aplica un 10%, de 3000€ a 6000€ s'aplica 13% i per damunt de 6000€ s'aplica un 20%.

La línia de valors per a la compra es talla en trams en els valors que apareixen en l'enunciat (300, 600, 1000, 3000, 6000), el valor de tall pertany al tram superior (a 300€ se li aplica un 5%, no el 0%).

Són trams de valors reals (hi ha infinits valors en cadascun d'ell), no es pot usar una selecció múltiple.

Els SI fan una pregunta sobre cada valor de tall (300, 600, ... 6000). Aquest SI estan niats, es pregunta pel valor de tall de forma ordenada, es pot començar per 300 i augmentar, o per 6000 i disminuir, això es pot fer, ja que el conjunt de nombres està ordenat.

Calcula el descompte que es resta de l'import de la compra.

Visualitza l'import de la compra, el descompte i import final

```
Scanner lector = new Scanner(System.in);
lector.useLocale(Locale.UK);
System.out.print("Entra l'import de la compra: ");
double importCompra = lector.nextDouble();
double descompte;
if (importCompra < 300) {
   descompte = 0;
} else {
   if (importCompra < 600) {
      descompte = importCompra * 0.05;
   } else {
      if (importCompra < 1000) {
         descompte = importCompra * 0.08;
      } else {
         if (importCompra < 3000) {
            descompte = importCompra * 0.1;
         } else {
            if (importCompra < 6000) {
               descompte = importCompra * 0.13;
            } else {
               descompte = importCompra * 0.2;
            }
         }
      }
   }
}
double importFinal = importCompra - descompte;
System.out.printf("import compra = %.2f€\ndescompte = %.2f€\nimport final = %.2f€\n",
                  importCompra, descompte, importFinal);
```

Altra versió,

Informació a manejar

- l'import de la compra, és **un nombre real** (double), importCompra
- el percentatge de descompte de la compra, és **un nombre enter** (int), perDescompte

La línia de valors es recorre des del valor de tall més alt fins al més baix.

El descompte i l'import final són càlculs que es realitzen en el mètode printf que avalua les expressions i visualitza el resultat.

```
Scanner lector = new Scanner(System.in);
lector.useLocale(Locale.UK);
System.out.print("Entra l'import de la compra: ");
double importCompra = lector.nextDouble();
int perDescompte = 0;
if (importCompra >= 6000) {
   perDescompte = 20;
} else {
   if (importCompra >= 3000) {
      perDescompte = 13;
   } else {
      if (importCompra >= 1000) {
         perDescompte = 10;
      } else {
         if (importCompra >= 600) {
            perDescompte = 8;
         } else {
            if (importCompra >= 300) {
               perDescompte = 5;
            }
         }
      }
   }
}
System.out.printf("import compra = %.2f€\ndescompte = %.2f€\nimport final = %.2f€\n",
      importCompra,
      perDescompte * importCompra / 100.0,
      importCompra * (1 - perDescompte / 100.0));
```

En els càlculs s'escriu 100.0 com a literal, perquè la divisió tinga decimals.

exemple 012

Escriu un programa que calcula i visualitza la sensació tèrmica de fred o calor.
Llig de teclat la temperatura de l'aire en graus Celsius (T).
Si la temperatura està entre 15º i 26º, la sensació tèrmica és igual a la temperatura.
Si la temperatura és inferior a 15º, llavors llig la velocitat del vent en km per hora (V), i aplica la fórmula per a calcular la sensació tèrmica de fred (Stc)
$Stc = 13.1267 + 0.6215 * T - 11.37 * (1.5 * V)^{0.16} + 0.3965 * T * (1.5 * V)^{0.16}$
Si la temperatura és inferior a -50° o si la velocitat del vent és major que 100km/h o menor que 4km/h, llavors trau el missatge "No es pot calcular la sensació tèrmica".
Si la temperatura és superior a 26º, llavors llig el percentatge d'humitat relativa (HR) per al càlcul sols s'usa el valor que és un enter, i aplica la fórmula per a calcular la sensació tèrmica (Stc)
$Stc = -8.78469476 + 1.61139411 * T + 2.338548839 * HR - 0.14611605 * T * HR - 0.012308094 * T^2 - 0.016424828 * HR^2 + 0.002211732 * T^2 * HR + 0.00072546 * T * HR^2 - 0.000003582 * T^2 * HR^2$
Si la humitat relativa és inferior a 40%, llavors trau el missatge "No es pot calcular la sensació tèrmica".
El mètode Math.pow(a, b) de Java calcula a^b

De la lectura de l'enunciat separa

- les dades
  - la temperatura de l'aire en graus Celsius
  - la velocitat del vent en km per hora
  - la sensació tèrmica
  - el percentatge d'humitat relativa
- les accions
  - **calcula i visualitza** la sensació tèrmica de fred o calor
  - **llig** de teclat la temperatura de l'aire en graus Celsius
  - **llig** la velocitat del vent en km per hora
  - **llig** el percentatge d'humitat relativa
- les condicions
  - Si la temperatura està entre 15º i 26º
  - Si la temperatura és inferior a 15º
  - Si la temperatura és inferior a -50°
  - si la velocitat del vent és major que 100km/h o menor que 4km/h
  - Si la temperatura és superior a 26º
  - Si la humitat relativa és inferior a 40%

Defineix la informació a manejar

Informació a manejar

- temperatura de l'aire en graus Celsius, és **un nombre real** (double), temperatura, els graus Celsius s'expressen en els literals
- la velocitat del vent en km per hora, és **un nombre real** (double), velocitatVent, km per hora s'expressa en els literals

- la humitat relativa, és **un nombre enter** (int), humitatRelativa, s'expressa en percentatges
- la sensació tèrmica, és **un nombre real** (double), sensacioTermica s'usa tant, per a la sensació tèrmica de calor o de fred

Defineix les accions

Llig de teclat la temperatura de l'aire en graus Celsius, temperatura

Has de comprovar quatre trams de temperatures

- inferior a -50ºC missatge "No es pot calcular la sensació tèrmica"
- inferior a 15ºC cal aplicar la fórmula de sensació tèrmica de fred
- entre 15ºC i 26ºC la sensació tèrmica és igual a la temperatura
- superior a 26ºC cal aplicar la fórmula de sensació tèrmica de calor

Calcula la sensació tèrmica de fred, has de llegir la velocitat del vent, si la velocitat del vent és major que 100km/h o menor que 4km/h, llavors trau el missatge "No es pot calcular la sensació tèrmica".

```
sensacioTermica = 13.1267 + 0.6215 * temperatura  - 11.37 * Math.pow(1.5 * velocitatVent,0.16)
                  + 0.3965 * temperatura * Math.pow(1.5 * velocitatVent,0.16);
```

Calcula la sensació tèrmica de calor, has de llegir la humitat relativa, si la humitat relativa és inferior a 40%, llavors trau el missatge "No es pot calcular la sensació tèrmica".

```
sensacioTermica =  -8.78469476 + 1.61139411 * temperatura + 2.338548839 * humitatRelativa
                   - 0.14611605 * temperatura * humitatRelativa
                   - 0.012308094 * Math.pow(temperatura, 2)
                   - 0.016424828 * Math.pow(humitatRelativa,2)
                   + 0.002211732 * Math.pow(temperatura, 2) * humitatRelativa
                   + 0.00072546 * temperatura * Math.pow(humitatRelativa,2)
                   - 0.000003582 * Math.pow(temperatura, 2) * Math.pow(humitatRelativa,2);;
```

Visualitza la sensació tèrmica.

```
Scanner lector = new Scanner(System.in);
lector.useLocale(Locale.UK);
System.out.print("entra la temperatura (ºC): ");
double temperatura = lector.nextDouble();
if (temperatura<-50) {
   System.out.println("No  pot calcular la sensació tèrmica");
} else {
   if (temperatura >= 15 && temperatura<=26) {
      System.out.printf("la sensació tèrmica és %.2fºC", temperatura);
   } else {
      if (temperatura< 15) {            //aplicar la fórmula de sensació tèrmica de fred
         System.out.print("entra la velocitat del vent (km/h): ");
         double velocitatVent = lector.nextDouble();
```

```
        if (velocitatVent>100 || velocitatVent<4) {
            System.out.println("No  pot calcular la sensació tèrmica");
        } else {
            double sensacioTermica = 13.1267 + 0.6215 * temperatura
                - 11.37 * Math.pow(1.5 * velocitatVent,0.16)
                + 0.3965 * temperatura * Math.pow(1.5 * velocitatVent,0.16);
            System.out.println("sensació tèrmica: "+sensacioTermica +" ºC");
        }
    } else {                        //aplicar la fórmula de sensació tèrmica de calor
        System.out.print("entra la humitat relativa: ");
        int humitatRelativa = lector.nextInt();
        if (humitatRelativa<40) {
            System.out.println("No  pot calcular la sensació tèrmica");
        } else {
            double sensacioTermica =
                -8.78469476 + 1.61139411 * temperatura + 2.338548839 * humitatRelativa
                - 0.14611605 * temperatura * humitatRelativa
                - 0.012308094 * Math.pow(temperatura, 2)
                - 0.016424828 * Math.pow(humitatRelativa,2)
                + 0.002211732 * Math.pow(temperatura, 2) * humitatRelativa
                + 0.00072546 * temperatura * Math.pow(humitatRelativa,2)
                - 0.000003582 * Math.pow(temperatura, 2) * Math.pow(humitatRelativa,2);
            System.out.println("sensació tèrmica: "+sensacioTermica +" ºC");
        }
    }
  }
}
```

Amb mètodes

getSensacioFred

El mètode permet obtindre la sensació tèrmica de fred d'una temperatura.

private static String getSensacioFred(double temperatura, double velocitatVent) {

- rep la temperatura, que és un double, temperatura (paràmetre)
- rep la velocitat del vent, que és un double, velocitatVent (paràmetre)
- torna una cadena de text
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- temperatura de l'aire en graus Celsius, és **un nombre real** (double), temperatura, els graus Celsius s'expressen en els literals
- la velocitat del vent en km per hora, és **un nombre real** (double), velocitatVent,
- la sensació tèrmica, és **un nombre real** (double), sensacioTermica, són graus Celsius s'expressen en els literals

Defineix les accions

Rep el double temperatura, la temperatura de l'aire en graus Celsius, i el double velocitatVent, la velocitat del vent en km per hora.

Si la velocitat del vent és major que 100km/h o menor que 4km/h, llavors torna el text "No es pot calcular la sensació tèrmica".

Calcula la sensació tèrmica de fred

```
sensacioTermica = 13.1267 + 0.6215 * temperatura  - 11.37 * Math.pow(1.5 * velocitatVent,0.16)
                        + 0.3965 * temperatura * Math.pow(1.5 * velocitatVent,0.16);
```

Retorna una cadena de text amb el format "sensació tèrmica: sensacioTermica ºC".

```
private static String getSensacioFred(double temperatura, double velocitatVent) {
  if (velocitatVent > 100 || velocitatVent < 4) {
     return "No es pot calcular la sensació tèrmica";
  }
  double sensacioTermica = 13.1267 + 0.6215 * temperatura
       - 11.37 * Math.pow(1.5 * velocitatVent, 0.16)
       + 0.3965 * temperatura * Math.pow(1.5 * velocitatVent, 0.16);
  return "sensació tèrmica: " + sensacioTermica + " ºC";
}
```

getSensacioCalor

El mètode permet obtindre la sensació tèrmica de calor d'una temperatura.

private static String getSensacioCalor(double temperatura, int humitatRelativa) {

- rep la temperatura, que és un double, temperatura, (paràmetre)
- rep la humitat relativa, que és un enter, humitatRelativa, (paràmetre)
- torna una cadena de text
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- temperatura de l'aire en graus Celsius, és **un nombre real** (double), temperatura, els graus Celsius s'expressen en els literals
- la humitat relativa, és **un nombre enter** (int), humitatRelativa, s'expressa en percentatges
- la sensació tèrmica, és **un nombre real** (double) amb la referència sensacioTermica, són graus Celsius s'expressen en els literals

Defineix les accions

Rep el double temperatura, la temperatura de l'aire en graus Celsius, i l'enter humitatRelativa, la humitat relativa en percentatges.

Si la humitat relativa és inferior a 40%, llavors retorna la cadena de text "No es pot calcular la sensació tèrmica".

```
sensacioTermica =  -8.78469476 + 1.61139411 * temperatura + 2.338548839 * humitatRelativa
                   - 0.14611605 * temperatura * humitatRelativa
                   - 0.012308094 * Math.pow(temperatura, 2)
                   - 0.016424828 * Math.pow(humitatRelativa,2)
                   + 0.002211732 * Math.pow(temperatura, 2) * humitatRelativa
                   + 0.00072546 * temperatura * Math.pow(humitatRelativa,2)
                   - 0.000003582 * Math.pow(temperatura, 2) * Math.pow(humitatRelativa,2);;
```

Retorna una cadena de text amb el format "sensació tèrmica: sensacioTermica ºC".

```
private static String getSensacioCalor(double temperatura, int humitatRelativa) {
   if (humitatRelativa < 40) {
      return "No  pot calcular la sensació tèrmica";
   }
   double sensacioTermica
         = -8.78469476 + 1.61139411 * temperatura + 2.338548839 * humitatRelativa
         - 0.14611605 * temperatura * humitatRelativa
         - 0.012308094 * Math.pow(temperatura, 2)
         - 0.016424828 * Math.pow(humitatRelativa, 2)
         + 0.002211732 * Math.pow(temperatura, 2) * humitatRelativa
         + 0.00072546 * temperatura * Math.pow(humitatRelativa, 2)
         - 0.000003582 * Math.pow(temperatura, 2) * Math.pow(humitatRelativa, 2);
   return "sensació tèrmica: " + sensacioTermica + " °C";
}
```

main

```
Scanner lector = new Scanner(System.in);
lector.useLocale(Locale.UK);
System.out.print("entra la temperatura (°C): ");
double temperatura = lector.nextDouble();
if (temperatura < -50) {
   System.out.println("No  pot calcular la sensació tèrmica");
} else {
   if (temperatura >= 15 && temperatura <= 26) {
      System.out.printf("la sensació tèrmica és %.2f°C", temperatura);
   } else {
      if (temperatura < 15) {              //aplica la fórmula de sensació tèrmica de fred
         System.out.print("entra la velocitat del vent (km/h): ");
         double velocitatVent = lector.nextDouble();
         System.out.println(getSensacioFred(temperatura, velocitatVent));
      } else {                             //aplica la fórmula de sensació tèrmica de calor
         System.out.print("entra la humitat relativa: ");
         int humitatRelativa = lector.nextInt();
         System.out.println(getSensacioCalor(temperatura, humitatRelativa));
      }
```

```
  }
}
```

exemple 013

Escriu un programa que augmenta el sou d'un empleat. Si pertany a la categoria 1, puja un 8.5%, a les categories 5 i 8 puja un 8%, a les categories 2, 3 i 4 puja un 7.8%, a la categoria 10 puja un 7.5% i a les altres categories puja un 6.8%. visualitza el sou final.

De la lectura de l'enunciat separa

- les dades
  - el sou d'un empleat
  - la categoría
  - el sou final
- les accions
  - **augmenta** el sou d'un empleat
  - **visualitza** el sou final
- les condicions
  - Si pertany a la categoria 1, puja un 8.5%, a les categories 5 i 8 puja un 8%, a les categories 2, 3 i 4 puja un 7.8%, a la categoria 10 puja un 7.5% i a les altres categories puja un 6.8%.

Defineix la informació a manejar

- el sou i el sou final, és el mateix concepte en dos moments diferents, són diners (euros), per tant, és un **nombre real** (double), sou
- la categoria, és **un nombre enter** (int), categoria
- l'augment a aplicar, és un **nombre real** (double), augment

Defineix les accions

Llig de teclat el sou inicial del treballador, sou i la categoria categoria

Una selecció múltiple sobre la categoria defineix el percentatge a usar en el càlcul.

El càlcul de l'augment realitza amb l'expressió: augment = sou * percentatge / 100;

Visualitza el sou final sou + augment

```
Scanner lector = new Scanner(System.in);
lector.useLocale(Locale.UK);
System.out.print("sou: ");
double sou = lector.nextDouble();
System.out.print("categoria: ");
int categoria = lector.nextInt();
double augment;
```

```
switch (categoria) {
   case 1:
      augment = sou * 8.5 / 100;
      break;
   case 5:
   case 8:
      augment = sou * 8.0 / 100;
      break;
   case 2:
   case 3:
   case 4:
      augment = sou * 7.8 / 100;
      break;
   case 10:
      augment = sou * 7.5 / 100;
      break;
   default:
      augment = sou * 6.8 / 100;
}
System.out.printf("sou final: %.2f€\n", sou + augment);
```

Usant notació lambda com una expressió per al switch

```
Scanner lector = new Scanner(System.in);
lector.useLocale(Locale.UK);
System.out.print("sou: ");
double sou = lector.nextDouble();
System.out.print("categoria: ");
int categoria = lector.nextInt();
double augment;
augment = switch (categoria) {
   case 1 ->        sou * 8.5 / 100;
   case 5, 8 ->     sou * 8.0 / 100;
   case 2, 3, 4 -> sou * 7.8 / 100;
   case 10 ->       sou * 7.5 / 100;
   default ->       sou * 6.8 / 100;
};
System.out.printf("sou final: %.2f€\n", sou + augment);
```

Usant la notació lambda, en aquest cas el switch proporciona el percentatge d'augment per al càlcul, no la quantitat de diners. La variable augment no és necessària.

```
Scanner lector = new Scanner(System.in);
lector.useLocale(Locale.UK);
System.out.print("sou: ");
double sou = lector.nextDouble();
System.out.print("categoria: ");
int categoria = lector.nextInt();
```

```
sou = sou + sou *  switch (categoria) {
   case 1 ->        8.5;
   case 5, 8 ->     8.0;
   case 2, 3, 4 -> 7.8;
   case 10 ->       7.5;
   default ->       6.8;
} / 100;
System.out.printf("sou final: %.2f€\n", sou);
```

exemple 014

Escriu un programa que visualitza el tipus de triangle i la seua superfície. Es lligen els tres costats del triangle. La fórmula de la superfície és

$$\text{sup} = \frac{1}{4} * \sqrt{(\text{costat1}^2 + \text{costat1}^2 + \text{costat1}^2)^2 - 2 * (\text{costat1}^4 + \text{costat2}^4 + \text{costat3}^4)}$$

Si el contingut de l'arrel quadrada no és positiu, llavors no és un triangle correcte.
Un triangle equilàter té els tres costats d'una mateixa longitud.
Un triangle isòsceles té els dos costats d'una mateixa longitud.
Un triangle escalé té els tres costats de longitud diferent.

De la lectura de l'enunciat separa

- les dades
  - el tipus de triangle i la seua superfície
  - els tres costats del triangle
- les accions
  - **visualitza** el tipus de triangle i la seua superfície
  - **lligen** els tres costats del triangle
- les condicions
  - Si el contingut de l'arrel quadrada no és positiu, llavors no és un triangle correcte.
  - Un triangle equilàter té els tres costats d'una mateixa longitud.
  - Un triangle isòsceles té els dos costats d'una mateixa longitud.
  - Un triangle escalé té els tres costats de longitud diferent.

Defineix la informació a manejar

- els tres costats del triangle, és el concepte costat repetits tres vegades, són tres **nombres reals** (double), costat1, costat2 i costat3
- la superfície del triangle, és un **nombre real** (double), superfície
- els tipus de triangle: "equilàter", "isòsceles" i "escalè" són tres literals

Defineix les accions

Llig de teclat els tres costats del triangle costat1, costat2 i costat3 .

El càlcul de la superfície d'un triangle usa la fórmula

$$\text{sup} = \frac{1}{4} * \sqrt{(\text{costat1}^2 + \text{costat1}^2 + \text{costat1}^2)^2 - 2 * (\text{costat1}^4 + \text{costat2}^4 + \text{costat3}^4)}$$

Necessites una variable auxiliar per a emmagatzemar el contingut de l'arrel quadrada, és un nombre real representat per aux, el mètode Math.pow(a,b) permet calcular potències.

$$\text{aux} = (\text{costat1}^2 + \text{costat1}^2 + \text{costat1}^2)^2 - 2 * (\text{costat1}^4 + \text{costat2}^4 + \text{costat3}^4)$$

Visualitza el missatge "Dades del triangle incorrectes" si el valor de aux és negatiu, això vol dir que els valors dels costats del triangle no són correctes.

Si el valor de aux és major que 0, llavors calcula la superfície, el mètode Math.sqrt(a) permet calcular arrels quadrades.

Compara els valors dels costats, segon el nombre de costats iguals visualitza el tipus de triangle i la superfície. Un triangle equilàter té els tres costats d'una mateixa longitud. Un triangle isòsceles té els dos costats d'una mateixa longitud. Un triangle escalé té els tres costats de longitud diferent.

Visualitza el tipus i la superfície.

```
Scanner lector = new Scanner(System.in);
lector.useLocale(Locale.UK);
System.out.println("entra la longitud dels costats del triangle");
System.out.print("costat1: ");
double costat1 = lector.nextDouble();
System.out.print("costat2: ");
double costat2 = lector.nextDouble();
System.out.print("costat3: ");
double costat3 = lector.nextDouble();
double aux = Math.pow(costat1 * costat1 + costat2 * costat2 + costat3 * costat3, 2)
      - 2 * (Math.pow(costat1, 4) + Math.pow(costat2, 4) + Math.pow(costat3, 4));
if (aux < 0) {
   System.out.println("Dades del triangle incorrectes");
} else {
   double superficie = Math.sqrt(aux) / 4;
   if (costat1 == costat2 && costat2 == costat3) {          // els tres costats iguals, equilàter
      System.out.println("el triangle és equilàter\nsuperfície = " + superficie);
   } else {
      if (costat1 == costat2 || costat2 == costat3 || costat1 == costat3) {        // dos costats iguals, isòsceles
         System.out.println("el triangle és isòsceles\nsuperfície = " + superficie);
      } else {          // els tres costats diferents, escalé
         System.out.println("el triangle és escalé\nsuperfície = " + superficie);
      }
   }
}
```

Amb mètodes

getSuperficie

El mètode permet obtindre la superfície d'un triangle:

private static double getSuperficie(double costat1, double costat2, double costat3) {

- rep els costats del triangle, tres double, costat1, costat2, costat3 (paràmetres)
- torna un double, si el valor és negatiu això vol dir que les dades dels costats són errònies
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- els costats del triangle, és el mateix concepte, són tres **nombres reals** (double), costat1, costat2 i costat3
- la superfície del triangle, és un **nombre real** (double), superfície

Defineix les accions

Rep costat1, costat2 i costat3, tres double, els costats del triangle.

El càlcul de la superfície d'un triangle és

$$\text{sup} = \frac{1}{4} * \sqrt{(\text{costat1}^2 + \text{costat1}^2 + \text{costat1}^2)^2 - 2 * (\text{costat1}^4 + \text{costat2}^4 + \text{costat3}^4)}$$

Necessites una variable auxiliar per a emmagatzemar el contingut de l'arrel quadrada, és un nombre real representat per aux.

Si aux és negatiu, llavors retorna el valor de aux.

Si aux és positiu o zero, retorna la superfície del triangle.

```
private static double getSuperficie(double costat1, double costat2, double costat3) {
   double aux = Math.pow(costat1 * costat1 + costat2 * costat2 + costat3 * costat3, 2)
                     - 2 * (Math.pow(costat1, 4) + Math.pow(costat2, 4) + Math.pow(costat3, 4));
   return aux < 0 ? aux : Math.sqrt(aux) / 4;
}
```

getTipus

El mètode permet obtindre el tipus de triangle:

private static String getTipus(double costat1, double costat2, double costat3) {

- rep els costats del triangle, tres double, costat1, costat2, costat3 (paràmetres)
- torna una cadena de text
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- els costats del triangle, és el mateix concepte, són tres **nombres reals** (double), costat1, costat2 i costat3
- els tipus de triangle: "equilàter", "isòsceles" i "escalè" són tres literals

Defineix les accions

Rep tres double costat1, costat2 i costat3, els costats del triangle.

Si els tres costats són iguals, retorna la cadena de text "equilàter"

Si dos costats són iguals, retorna la cadena de text "isòsceles"

Si els tres costats són diferents, retorna la cadena de text "escalé"

```
private static String getTipus(double costat1, double costat2, double costat3) {
  if (costat1 == costat2 && costat2 == costat3) {          // els tres costats iguals, equilàter
    return "equilàter";
  }
  if (costat1 == costat2 || costat2 == costat3 || costat1 == costat3) {          // dos costats iguals, isòsceles
    return "isòsceles";
  }       // els tres costats diferents, escalé
  return "escalé";
}
```

main

```
Scanner lector = new Scanner(System.in);
lector.useLocale(Locale.UK);
System.out.println("entra la longitud dels costats del triangle");
System.out.print("costat1: ");
double costat1 = lector.nextDouble();
System.out.print("costat2: ");
double costat2 = lector.nextDouble();
System.out.print("costat3: ");
double costat3 = lector.nextDouble();
double superficie = getSuperficie(costat1, costat2, costat3);
if (superficie < 0) {
  System.out.println("Dades del triangle incorrectes");
} else {
  System.out.println("el triangle és " + getTipus(costat1, costat2, costat3)+ "\nsuperfície = " + superficie);
}
```

exemple 024

Escriu un programa que visualitza els 100 primers nombres enters.

De la lectura de l'enunciat separa

- les dades
  - 100 primers nombres enters
- les accions
  - **visualitza** els 100 primers nombres enters
- les condicions
  - els 100 primers nombres enters

Defineix la informació a manejar

- 100 nombres enters, és el mateix concepte, **un nombre enter** (int), num

Defineix les accions

L'única acció a realitzar és visualitzar, des d'1 a 100, la dada a visualitzar és un nombre enter que ha d'adquirir aquests valors.

El bucle es repeteix 100 vegades, en cada iteració el nombre té un valor diferent. La variació del valor de l'enter serveix per a controlar el bucle. El valor inicial del nombre és 1, el nombre s'incrementa d'1 en 1 (variació que afecta algun element de l'expressió del bucle) i la condició de final és mentre el nombre és menor o igual a 100

```
int num = 1;                        // inicialització a 1
while (num <= 100) {                // mentre num menor o igual que 100
   System.out.print(num + "  ");    // visualitza num
   num++;                           // variació, suma 1 a num
}
```

exemple 026

Escriu un programa que calcula el punt quilomètric on es creuen dos cotxes que viatgen per l'autovia A3 (Madrid - València) i el visualitza.
El cotxe un ix del complex educatiu de Xest en el quilòmetre 333 i va cap a Madrid, el cotxe dos ix de Tébar en el quilòmetre 186 i va cap a València. Els dos cotxes van a la mateixa velocitat. Usa un bucle que para quan es creuen els vehicles.
Fes dues versions, on els cotxes van a la mateixa velocitat, amb una variació sencera i amb una variació real.
Fes una tercera versió, on el cotxe un va més de pressa que el cotxe dos (variació real).

De la lectura de l'enunciat separa

- les dades
  - el punt quilomètric
  - el quilòmetre del cotxe que va cap a Madrid
  - el quilòmetre del cotxe que va cap a València
  - velocitat

- les accions
    - **calcula** el punt quilomètric on es creuen dos cotxes
    - **visualitza** el punt quilomètric on es creuen dos cotxes
- les condicions
    - para quan es creuen els vehicles

Defineix la informació a manejar

primera versió

- la posició cotxe que va cap a Madrid, és **un nombre enter** (int), posAM
- la posició cotxe que va cap a València, és **un nombre enter** (int), posAV
- la velocitat dels cotxes, és **un literal enter** que val 1

Defineix les accions

El punt quilomètric correspon a la posició d'uns dels cotxes quan es creuen.

En el bucle incrementa la posició del cotxe que va cap a València i es decrementa la posició del cotxe que va cap a Madrid, mentre la posició del cotxe que va cap a València és menor o igual que la posició del cotxe que va cap a Madrid.

La velocitat és la mateixa, el literal és 1, s'augmenta o es disminueix la posició del cotxe en 1 quilòmetre.

```
int posAM = 333;
int posAV = 186;
while (posAM >= posAV) {
   posAM--;          // variació de la posició menys 1
   posAV++;          // variació de la posició més 1
}
System.out.println("s'han creuat en  " + posAM + "  " + posAV);
```

segona versió

- la posició cotxe que va cap a Madrid, és **un nombre real** (double), posAM
- la posició cotxe que va cap a València, és **un nombre real** (double), posAV
- la velocitat dels cotxes, és un literal real que val 0.25

La velocitat és la mateixa, el literal és 0.25, s'augmenta o es disminueix la posició del cotxe en 0.25 quilòmetre.

```
double posAM = 333;
double posAV = 186;
final double velocitat = 0.25;          // la velocitat és 0.25, la mateixa per als dos cotxes
while (posAM >= posAV) {
   posAM -= velocitat;          // variació de la posició menys la velocitat
   posAV += velocitat;          // variació de la posició més la velocitat
}
System.out.println("s'han creuat en  " + posAM + "  " + posAV);
```

tercera versió

- la posició cotxe que va cap a Madrid, és **un nombre real** (double), posAM
- la posició cotxe que va cap a València, és **un nombre real** (double), posAV
- la velocitat del cotxe que va cap a Madrid, és **un nombre real** (double), velocitatAM, és una constant
- la velocitat del cotxe que va cap a València, és **un nombre real** (double), velocitatAV2, és una constant

Els dos cotxers van a velocitats diferents, però constants.

```
double posAM = 333;
double posAV = 186;
final double velocitatAM = 0.45;        // la velocitat és 0.45 per al cotxe més ràpid
final double velocitatAV = 0.36;        // la velocitat és 0.36 per al cotxe més lent
while (posAM >= posAV) {
   posAM -= velocitatAM;     // variació de la posició menys la velocitatAM
   posAV += velocitatAV;     // variació de la posició més la velocitatAV
}
System.out.println("s'han creuat en  " + posAM + "  " + posAV);
```

exemple 015

Escriu un programa que llig de teclat 10 nombres reals i visualitza el nombre màxim i mínim.

De la lectura de l'enunciat separa

- les dades
  - 10 nombres reals
  - el nombre màxim i mínim
- les accions
  - **llig** de teclat 10 nombres reals
  - **visualitza** el nombre màxim i mínim
- les condicions
  - repetir 10 lectures de nombres
  - màxim i mínim

Defineix la informació a manejar

- un nombre real, és **un nombre real** (double), num
- el nombre màxim, és **un nombre real** (double), max
- el nombre mínim, és **un nombre real** (double), min
- el comptador del bucle de lectura, és **un nombre enter** (int), cont

Defineix les accions

Llig de teclat els 10 nombres, has d'usar un bucle que es repeteix 10 vegades, necessites un comptador, és un nombre enter representat per cont.

Els nombres sols es poden comparar a partir del segon nombre, per això el primer nombre és, tant, el màxim com el mínim.

La primera lectura és especial i es realitza fora del bucle, inicia el comptador de nombres a 1 (ja has llegit un nombre).

En el bucle llig num, si num és major que el nombre major max, llavors canvia el nombre major, i si num és menor que el nombre menor min, llavors canvia el nombre menor.

Incrementa el comptador. El bucle es repeteix mentre cont < 10, són 9 iteracions, més la primera lectura sumen els 10 nombres a llegir.

Visualitza el nombre major max i el menor min.

```
Scanner lector = new Scanner(System.in);
System.out.println("entra 10 nombres enters");
System.out.print("0> entra num: ");
int num = lector.nextInt();                    // lectura del primer nombre
int max = num;                                 // el primer nombre és el màxim
int min = num;                                 // el primer nombre és el mínim
int cont = 1;                                  // ja has llegit un nombre, el comptes
while (cont < 10) {                            // mentre el comptador és menor que 10 repeteix el bucle
   System.out.print(cont + "> entra num: ");   // cont indica quin nombre estàs llegint
   num = lector.nextInt();                     // llig del nombre
   if (num > max) {                            // si el nombre és major que el màxim
      max = num;                               //  canvia el màxim
   }
   if (num < min) {                            // si el nombre és menor que el mínim
      min = num;                               //  canvia el mínim
   }
   cont++;                                     // incrementa el comptador
}
System.out.println("màxim = " + max + "  mínim = " + min);
```

exemple 016

Escriu un programa que llig de teclat el preu d'un article i l'IVA que cal aplicar-li (4, 10 o 21) i visualitza el seu preu de venda.
Si no es llig 4, 10 o 21 per a l'IVA, llavors es repeteix la lectura.

De la lectura de l'enunciat separa

- les dades
    - el preu d'un article
    - l'IVA que cal aplicar
- les accions
    - **llig** de teclat el preu d'un article i l'IVA
    - **visualitza** el seu preu de venda
- les condicions
    - si no es llig 4, 10 o 21 per a l'IVA

Defineix la informació a manejar

- el preu de l'article, és **un nombre real** (double), preu
- l'IVA a aplicar, és **un nombre enter** (int), iva
- el preu de venda de l'article, és **un nombre real** (double), pvp

Defineix les accions

Llig de teclat el preu.

Llig de teclat l'iva, has de realitzar un bucle de lectura per a obligar a l'usuari a introduir un valor correcte per a l'iva, és a dir, 4, 10 o 21. Repeteix el bucle de lectura mentre iva no siga 4 i iva no siga 10 i iva no siga 21 (iva != 4 && iva != 10 && iva != 21)

Usa un bucle amb la condició al final, la inicialització per a poder avaluar l'expressió és la lectura del valor de iva.

Amb aquest format, has de tindre en compte que l'expressió està fora del bloc de repetició, per tant, has de definir la variable iva abans del bucle, perquè siga accessible en el bloc i en l'expressió.

El càlcul del preu de venda és pvp = preu + preu * iva / 100

Visualitza el preu de venda

```
Scanner lector = new Scanner(System.in);
lector.useLocale(Locale.UK);
System.out.print("preu: ");
double preu = lector.nextDouble();
int iva;    // iva s'ha de definir fora del bucle, per a ser accessible en l'expressió
do {
   System.out.print("IVA (4, 10, 21): ");
   iva = lector.nextInt();       //  llig el valor a controlar en l'expressió
} while (iva != 4 && iva != 10 && iva != 21);         // mentre l'expressió siga verdadera,  repeteix el bloc del do
double pvp = preu + preu * iva / 100;
System.out.println("preu de venda = " + pvp);
```

exemple 025

Escriu un programa que llig nombres enters positius i compta els nombres parells i imparells. Després de la lectura del nombre, pregunta si volem llegir altre, si introduïm "S" o "s", llavors es llig altre nombre, sinó finalitza el bucle i es visualitza el nombre total de parells i d'imparells.

De la lectura de l'enunciat separa

- les dades
  - nombres enters positius
  - nombres parells i imparells
  - si introduïm "S" o "s"
- les accions
  - **llig** nombres enters positius
  - **compta** els nombres parells i imparells
  - **pregunta** si volem llegir altre
  - **visualitza** el nombre total de parells i d'imparells
- les condicions
  - enters positius
  - nombres parells i imparells
  - si volem llegir altre
  - si introduïm "S" o "s"

Defineix la informació a manejar

- un nombre enter, és **un nombre enter** (int), num
- nombre total de parells, és **un nombre enter** (int), totalParells. Un nombre és parell si és divisible per dos, és a dir, si num % 2 == 0 (mòdul 2 del nombre).
- nombre total d'imparells, és **un nombre enter** (int), totalImparells. Un nombre és imparell si no és parell.
- introduir "S" o "s" per a indicar que volem llegir altre nombre, és una **cadena de text** (String), sn

Defineix les accions

Els totals són acumuladors de sumes, per tant, s'inicialitzen a 0, prèviament al bucle.

El bucle es repeteix mentre sn siga igual a "S" o "s", posa String sn = "S"; prèviament al bucle per a assegurar l'entrada al bucle (l'expressió és vertadera). La condició de final del bucle és qualsevol valor de sn que no siga "S" o "s".

En el bucle, quan el nombre és negatiu no es fa el tractament del mateix i es torna a llegir.

```
Scanner lector = new Scanner(System.in);
int totalParells = 0;
int totalImparells = 0;
String sn = "S";
```

```
while (sn.equalsIgnoreCase("S")) {     // mentre sn igual a "S" sense diferenciar majúscules de minúscules
   System.out.print("num: ");
   int num = lector.nextInt();
   if (num>=0) {     // si num és positiu
      if (num%2==0) {          // si num és parell
         totalParells++;
      } else {          // si num és imparell
         totalImparells++;
      }
      lector.nextLine();          // per a buidar el buffer d'entrada, assegurar la lectura en el següent nextLine
      System.out.print("Vols introduir altre nombre (S/N): ");
      sn = lector.nextLine();
   }
}
System.out.println("Has llegit "+totalParells+" parells i " + totalImparells+" imparells");
```

La sentència lector.nextLine(); elimina el retorn de carro deixat per lector.nextInt() perquè sn = lector.nextLine(); funcione correctament i no salte la lectura.

```
Scanner lector = new Scanner(System.in);
int totalParells = 0;
int totalImparells = 0;
String sn = "S";
while (sn.equalsIgnoreCase("S")) {
   int num = 0;
   do {
      System.out.print("entra un nombre positiu: ");
      num = lector.nextInt();
   } while (num<0);          // bucle que obliga a llegir un nombre positiu
   if (num%2==0) {
      totalParells++;
   } else {
      totalImparells++;
   }
   lector.nextLine();          // per a buidar el buffer d'entrada
   do {
      System.out.print("Vols introduir altre nombre (S/N): ");
      sn = lector.nextLine();
   } while (!sn.equalsIgnoreCase("S") && !sn.equalsIgnoreCase("N"));          // bucle que obliga a llegir "S" o "N"
}
System.out.println("Has llegit "+totalParells+" parells i " + totalImparells+" imparells");
```

exemple 017

Escriu un programa que llig un valor entre 333 i 888. Si és parell, visualitza els 20 parells posteriors, si és imparell, visualitza els 15 imparells anteriors.

De la lectura de l'enunciat separa

- les dades
  - un valor entre 333 i 888
- les accions
  - **llig** un valor entre 333 i 888
  - **visualitza** els 20 parells posteriors
  - **visualitza** els 15 imparells anteriors
- les condicions
  - un valor entre 333 i 888
  - és parell
  - és imparell

Defineix la informació a manejar

- un valor, és **un nombre enter** (int) amb la referència num
- un comptador per a la visualització de 15 o 20, és **un nombre enter** (int), cont

Defineix les accions

Llig de teclat el nombre entre 333 i 888, és un bucle que es repeteix mentre el nombre llegit és menor que 333 o major que 888, usa un bucle amb la condició al final.

Si el nombre és parell (num % 2 == 0), llavors el comptador val 20, visualitza el nombre i per a passar d'un parell al següent suma 2 al nombre.

Si el nombre és imparell (num % 2 == 1), llavors el comptador val 15, visualitza el nombre i per a passar d'un imparell al següent resta 2 al nombre.

No es fan les dues preguntes, un nombre és parell o imparell, es fa una única pregunta i s'usa el sinó.

```
Scanner lector = new Scanner(System.in);
int num;
do {
   System.out.print("Entra un nombre entre 333 i 888: ");
   num = lector.nextInt();
} while (num < 333 || num > 888);     //  repeteix mentre num menor que 333 o num major que 888
if (num % 2 == 0) {                   //  pregunta si num és parell
   for (int i = 0; i < 20; i++) {     // bucle que repeteix 20 vegades
      System.out.print(num+", ");
      num += 2;                       // afegeix 2 a num, com és parell al sumar-li 2 passa al parell següent
   }
} else {
   for (int i = 0; i < 15; i++) {     // bucle que  repeteix 15 vegades
```

```
    System.out.print(num+", ");
    num -= 2;                          // resta 2 a num, com és imparell al restar-li 2 passa a l'imparell anterior
  }
}
```

Amb mètodes

lligEntre

El mètode permet llegir un enter dins d'un rang

private static int lligEntre(int inf, int sup)

- rep el valor inferior i el superior, inf, sup són dos enters (paràmetres)
- torna un enter N que compleix inf <= N <= sup
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- els valors inferior i superior (int), inf i sup
- el valor, és un **nombre enter** (int), num

Defineix les accions

Rep dos enters inf i sup, comprova si el valor inferior és realment inferior al superior, si no és així, s'intercanvia els valors.

Repeteix la lectura del valor mentre el valor està fora del rang definit per inf i sup.

Retorna el valor llegit num.

```
private static int lligEntre(int inf, int sup) {
  Scanner lector = new Scanner(System.in);
  if (inf > sup) {
    int tmp = inf;
    inf = sup;
    sup = tmp;
  }
  int num;
  do {
    System.out.print("Entra un nombre entre " + inf + " i " + sup + ": ");
    num = lector.nextInt();
  } while (num < inf || num > sup);
  return num;
}
```

esParell

El mètode indica si un enter és parell

private static boolean esParell(int num)

- rep un valor enter, num (paràmetre)
- torna true si num és parell o false si num és imparell
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- el valor, és un **nombre enter** (int), num

Defineix les accions

Rep el valor enter num.

Retorna el resultat de num % 2 == 0., si num és divisible per 2 el resultat és 0, per tant és parell.

```
private static boolean esParell(int num) {
   return num % 2 == 0;
}
```

veureParells

El mètode mostra 20 parells superiors a un nombre

private static void veureParells(int num)

- rep un valor enter, num
- visualitza els 20 parells superiors a num
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- el valor, és un **nombre enter** (int), num
- el literal 20

Defineix les accions

Rep el valor enter num (és un nombre parell).

El bucle funciona sobre el valor del literal 20.

Visualitza num.

La variació de num és de 2 i sumant, per a passar d'un valor parell al següent.

No retorna res.

```
private static void veureParells(int num) {
   for (int i = 0; i < 20; i++) {
      System.out.print(num + ", ");
```

```
    num += 2;
  }
}
```

veureImparells

El mètode mostra 15 imparells inferiors a un nombre:

private static void veureImparells(int num) {

- rep un valor enter, num
- visualitza els 15 imparells inferiors a num
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- el valor, és un **nombre enter** (int), num
- el literal 15

Defineix les accions

Rep el valor enter num (és un nombre imparell).

El bucle funciona sobre el valor del literal 15

Visualitza num.

La variació de num és és de 2 i restant, per a passar d'un valor imparell a l'anterior.

No retorna res.

```
private static void veureImparells(int num) {
  for (int i = 0; i < 15; i++) {
    System.out.print(num + ", ");
    num -= 2;
  }
}
```

main

```
int num = lligEntre(333, 888);
if (esParell(num)) {
  veureParells(num);
} else {
  veureImparells(num);
}
```

exemple 028

Escriu un programa que llig un nombre entre 15 i 5555, i visualitza aquest nombre de "-".
Per a facilitar el recompte, el "-" en les posicions múltiples de 10 es canvia per "+", en les posicions múltiples de 100 per "/" i en les posicions múltiples de 1000 per "#".
Cada 100 caràcters es baixa de línia.

De la lectura de l'enunciat separa

- les dades
  - un nombre entre 15 i 5555
  - les posicions
- les accions
  - **llig** un nombre entre 15 i 5555
  - **visualitza** aquest nombre de "-"
  - **canvia** per "+", en les posicions múltiples de 100 per "/" i en les posicions múltiples de 1000 per "#"
- les condicions
  - n nombre entre 15 i 5555
  - les posicions múltiples de 10 es canvia per "+"
  - les posicions múltiples de 100 per "/"
  - les posicions múltiples de 1000 per "#"
  - Cada 100 caràcters es baixa de línia.

Defineix la informació a manejar

- un nombre, és **un nombre enter** (int), num
- la posició del "-", és **un nombre enter** (int), pos

Defineix les accions

Llig de teclat el nombre entre 15 i 5555, és un bucle que es repeteix mentre el nombre llegit és menor que 15 o major que 5555. Usa un bucle amb la condició al final.

La posició del "-" comença en 1 i s'incrementa d'un en un.

Per a comprovar el múltiple usa el mòdul, cal tindre en compte que una posició múltiple de 1000, també és múltiple de 100 i de 10. Has d'usar if niats i començar preguntant si la posició és múltiple de 1000, sinó si és múltiple de 100, sinó si és múltiple de 10 i substituir el "-" pel caràcter corresponent.

En els caràcters "#" i "/" s'afegeix un canvi de línia perquè són múltiples de 100.

En el bucle visualitza un caràcter en cada iteració el "-" o el caràcter corresponent si és múltiple de 10, 100 o 1000.

```
Scanner lector = new Scanner(System.in);
int num;
```

```
do {
   System.out.print("Entra un nombre entre 15 i 5555: ");
   num = lector.nextInt();
} while (num < 15 || num > 5555);          //  repeteix mentre num és incorrecte
int pos = 1;                               // posició del "-"
while (pos <= num) {
   String car = "-";
   if (pos % 1000 == 0) {
      car = "#\n";                         // afegeix un canvi de línia
   } else {
      if (pos % 100 == 0) {
         car = "/\n";                      // afegeix un canvi de línia
      } else {
         if (pos % 10 == 0) {
            car = "+";
         }
      }
   }
   System.out.print(car);                  // visualitza el caràcter
   pos++;                                  // canvia la posició
}
```

Amb mètodes

lligEntre

El mètode permet llegir un enter dins d'un rang.

És el mètode definit en l'exemple 017.

getCarEn

El mètode retorna el caràcter en una posició

private static String getCarEn(int pos)

- rep una posició, pos (paràmetre)
- torna el caràcter que ha d'apareixer en aquesta posició,
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- la posició, és un **nombre enter** (int), pos
- el caràcter, és un literal "#", "/", "+" i "-"

Defineix les accions

Rep el valor enter pos.

Si la posició és múltiple de 1000 torna "#\n", el canvi de línia per ser múltiple de 100.

Si la posició és múltiple de 100, el canvi de línia per ser múltiple de 100.

Si la posició és múltiple de 10

Retorna "-".

No és necessari el else, ja que el return finalitza el mètode.

```
private static String getCarEn(int pos) {
   if (pos % 1000 == 0) {
      return "#\n";
   }
   if (pos % 100 == 0) {
      return "/\n";
   }
   if (pos % 10 == 0) {
      return "+";
   }
   return "-";
}
```

exemple 018

Escriu un programa que visualitza la sèrie de valors d'ULAM. Llig de teclat un valor enter positiu major que 1. Aquest nombre és el primer de la sèrie. Si és parell, divideix-ho entre 2, si és imparell, multiplica-ho per 3 i suma-li 1, i visualitza el nou valor. Això es repeteix fins a obtindre un 1.

De la lectura de l'enunciat separa

- les dades
    - un valor enter positiu major que 1
- les accions
    - **visualitza** la sèrie de valors d'ULAM
    - **llig** de teclat un valor enter positiu major que 1
    - **visualitza** el nou valor
- les condicions
    - si és parell
    - si és imparell
    - fins a obtindre un 1

Defineix la informació a manejar

- un valor, és **un nombre enter** (int), num
- la sèrie de valors d'ULAM, és una sèrie de nombres enters que corresponen als diferents valors que adquireix num, no es guarden, es visualitzen en la consola

Defineix les accions

Llig de teclat el nombre inicial num. Ha de ser un nombre positiu major que 1, això és un bucle que es repeteix mentre el nombre llegit no siga correcte.

Visualitza el valor de num

El bucle es repeteix mentre num és major que 1, en el bucle es fa el següent.

Si num és parell, és a dir, si num mod 2 = 0, llavors num = num / 2

Si num és imparell, és a dir, si num mod 2 != 0, llavors num = num * 3 +1

Un nombre només pot ser parell o imparell, per això, els dos SI anteriors s'ajunten en un.

Visualitza num

```
Scanner lector = new Scanner(System.in);
int num;
do {
   System.out.print("Entra un nombre major que 1: ");
   num = lector.nextInt();
} while (num <= 1);                  //  repeteix mentre num menor o igual que 1
System.out.print(num + ", ");
while (num > 1) {                    // bucle que es repeteix mentre num > 1
   if (num % 2 == 0) {               // si num és parell
      num = num / 2;
   } else {                          // si num és imparell
      num = num * 3 + 1;
   }
   System.out.print(num + ", ");
}
```

exemple 027

Escriu un programa que llig un nombre enter major que 100000 i visualitza les xifres que el componen a l'inrevés (125073 --> 370521), repeteix el procés fins a llegir un zero.

De la lectura de l'enunciat separa

- les dades
  - un nombre enter major que 100000
  - les xifres que el componen a l'inrevés
- les accions
  - **llig** un nombre enter major que 100000

    - **visualitza** les xifres que el componen a l'inrevés
    - **repeteix** el procés fins a llegir un zero
- les condicions
    - major que 100000
    - fins a llegir un zero

Defineix la informació a manejar

- un nombre enter, és **un nombre enter** (int), num
- les xifres que el componen, és una sèrie de nombres enters que corresponen a les diferents unitats que adquireix num, no es guarden, es visualitzen en la consola

Defineix les accions

Llig de teclat el nombre inicial num. Ha de ser un nombre positiu major que 100000, això és un bucle que es repeteix mentre el nombre llegit no siga correcte. Cal tindre en compte que el 0 és necessari per a finalitzar l'altre bucle, per tant, cal incluir les dues condicions en l'expressió del final del bucle de lectura.

```
int num;
do {
   System.out.print("Entra un nombre major que 100000 (0 acaba): ");
   num = lector.nextInt();
} while (num != 0 && num < 100000);
```

El bucle repeteix la lectura mentre num és diferent de 0 i menor que 100000. Si num és 0, llavors num != 0 és falsa i no s'avalua num < 100000.

La unitat d'un nombre s'obté aplicant-li el mòdul de 10 (10 és la base del nombre)

```
unitat = num % 10;
```

Has de visualitzar la xifra obtinguda, i eliminar-la del nombre, dividint el nombre per 10.

```
num = num / 10;
```

Repeteix aquests passos mentre el nombre és major que 0.

```
Scanner lector = new Scanner(System.in);
int num;
do {
   System.out.print("Entra un nombre major que 100000 (0 acaba): ");
   num = lector.nextInt();
} while (num != 0 && num < 100000);          //  repeteix mentre num és diferent de 0 i menor que 100000
while (num != 0) {
   while (num>0) {
      int unitat = num % 10;                 // obté la unitat de num
      System.out.print(""+unitat);           // visualitza la unitat
```

```
    num = num / 10;                          // Elimina la unitat de num
  }
  System.out.println("");
  do {
    System.out.print("Entra un nombre major que 100000 (0 acaba): ");
    num = lector.nextInt();
  } while (num != 0 && num < 100000);        //  repeteix mentre num és diferent de 0 i menor que 100000
}
```

Altra versió

```
Scanner lector = new Scanner(System.in);
int num;
do {
  System.out.print("Entra un nombre major que 100000 (0 acaba): ");
  num = lector.nextInt();
  if (num != 0 && num >= 100000) {  // si num és diferent de 0 i major o igual que 100000
    int copia = num;          //  fa una còpia de num, és la variable que  modificarà en el bucle
    while (copia > 0) {
      System.out.print("" + (copia % 10));
      copia /= 10;
    }
    System.out.println("");
  }
} while (num != 0);             //  repeteix mentre num és diferent de 0
```

Es fa una còpia de num, ja que l'expressió del bucle num != 0, necessita el valor de num, i aquest canvia en el bucle intern.

Amb mètodes

A continuació hi ha quatre versions del mètode que permet invertir els dígits d'un nombre.

inverteix

private static int inverteix(int num) {

- rep l'enter num (paràmetre)
- torna un enter reves amb els dígits de num al revés
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- el nombre, num
- la unitat del nombre, unitat
- el valor enter al revés, reves

Defineix les accions

Rep l'enter num,

Obté el dígit que és la unitat de num amb el mòdul 10.

Multiplica el valor de reves per 10 (és la base dels nombres decimal, passa de unitats a desenes a centenes a etc.)

La unitat que s'ha tractat s'elimina de num amb la divisió per 10.

Repeteix el procés mentre num és major que 0 .

Retorna el valor construït en reves.

```
private static int inverteix(int num) {
   int reves = 0;
   while (num > 0) {
      int unitat = num % 10;
      reves = reves * 10 + unitat;
      num /= 10;
   }
   return reves;
}
```

inverteixChar

El mètode inverteix permet invertir els dígits d'un enter:

- rep l'enter num (paràmetre)
- torna una cadena de text reves amb els caràcters dels dígits de num al revés
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- el nombre, num
- la unitat del nombre, unitat
- la cadena de text amb els dígits al revés, reves

Defineix les accions

Rep l'enter num.

Obté el dígit que és la unitat de num amb el mòdul 10 i el transforma a caràcter.

El caràcter es concatena a reves.

La unitat que s'ha tractat s'elimina de num amb la divisió per 10.

Repeteix el procés mentre num és major que 0 .

Retorna el valor construït en reves.

```
private static String inverteixChar(int num) {
   String reves = "";
   while (num > 0) {
      int unitat = num % 10;
      reves += unitat ;
      num /= 10;
   }
   return reves;
}
```

inverteixString

El mètode inverteix permet invertir els dígits d'un enter:

- rep l'enter num (paràmetre)
- torna una cadena de text reves amb els caràcters dels dígits de num al revés
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- el nombre, num
- la cadena de text, cad
- la cadena de text amb els dígits al revés, reves

Defineix les accions

Rep l'enter num.

Crea la cadena de text cad amb el valor de num.

Recorre la cadena de text, començant pel final (i = cad.length()).

Obté el dígit (una cadena de text) que està en la posició i, amb substring(i-1, i).

El dígit es concatena a reves.

Repeteix l'obtenció de les subcadenes fins arribar a la posició 0 .

Retorna el valor construït en reves.

```
private static String inverteixString(int num) {
   String cad = num+"";
   String reves = "";
   for (int i = cad.length(); i>0; i--) {
      reves+=cad.substring(i-1, i);
   }
   return reves;
}
```

inverteixStringBuilder

El mètode inverteixStringBuilder permet invertir els dígits d'un enter:

- rep l'enter num (paràmetre)
- torna una cadena de text reves amb els caràcters dels dígits de num al revés
- és un mètode privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- el nombre, num
- el StringBuilder, cad

Defineix les accions

Rep l'enter num.

Crea el StringBuilder cad amb el valor de num transformat a cadena de text.

Aplica el mètode reverse per a invertir els caràcters que formen cad.

Retorna el valor toString de cad.

```
private static String inverteixStringBuilder(int num) {
   StringBuilder cad =  new StringBuilder(num+"");
   return cad.reverse().toString();
}
```

main

```
Scanner lector = new Scanner(System.in);
int num;
do {
   System.out.print("Entra un nombre major que 100000 (0 acaba): ");
   num = lector.nextInt();
   if (num != 0 && num >= 100000) {
      System.out.println("al revés és " + inverteix(num));
      System.out.println("al revés és " + inverteixChar(num)+" char");
      System.out.println("al revés és " + inverteixString(num)+" String");
      System.out.println("al revés és " + inverteixStringBuilder(num)+" StringBuilder");
   }
} while (num != 0);
```

En el main es criden els quatre mètodes.

Comprova la diferència de comportament dels mètodes que usen nombres i operacions aritmètiques o dels que usen caràcters (prova el 124100)

exemple 019

Escriu un programa que visualitza tots els nombres de tres dígits que es poden crear amb els dígits 1, 2, 3, 4 i 5, sense repetir. La visualització es realitza escrivint 10 nombres per línia. Al final visualitza quants nombres s'han visualitzat.

De la lectura de l'enunciat separa

- les dades
  - els nombres de tres dígits
  - els dígits 1, 2, 3, 4 i 5
  - quants nombres s'han visualitzat
- les accions
  - **visualitza** tots els nombres de tres dígits
  - **crear** amb els dígits 1, 2, 3, 4 i 5
  - **visualitza** quants nombres s'han visualitzat
- les condicions
  - nombres de tres dígits que es poden crear amb els dígits 1, 2, 3, 4 i 5, sense repetir
  - 10 nombres per línia

Defineix la informació a manejar

- els dígits 1, 2, 3, 4 i 5, són **nombres enters** (int), necessiten tres dígits per a construir cada nombre, d1, d2 i d3
- els nombres de tres dígits no es guarden, es creen amb d1, d2 i d3 i es mostren
- el comptador de nombres, és **un nombre enter** (int), comptador, el comptador s'usa també per a escriure 10 nombres per línia

Defineix les accions

El comptador de nombres és un acumulador de sumes, s'inicialitza amb 1 i s'incrementa d'un en un.

Hi ha tres bucles, un per a cada dígit (d1, d2 i d3) del nombre a crear, i estan niats. Per a cada dígit realitza un bucle que recorre els diferents valors que pot adquirir el dígit, és a dir, un bucle de l'1 al 5.

El bucle més intern comprova que no hi ha cap dígit repetit, i visualitza el nombre creat amb la concatenació dels tres dígits i el compta. La comprovació dels dígits repetits és, si d1 != d2 && d2 != d3 && d1 != d3, llavors el nombre és correcte, es mostra i es compta.

Si el comptador és un múltiple de 10, llavors es baixa de línia, és a dir, s'escriu un "\n", això és per a facilitar la lectura del resultat.

```
int comptador = 0;
for (int d1 = 1; d1 <= 5; d1++) {                    // bucle per al primer dígit
  for (int d2 = 1; d2 <= 5; d2++) {                  // bucle per al segon dígit
```

```
    for (int d3 = 1; d3 <= 5; d3++) {                    // bucle per al tercer dígit
      if (d1 != d2 && d2 != d3 && d1 != d3) {
        System.out.print("" + d1 + d2 + d3 + ", ");
        comptador++;
        if (comptador % 10 == 0) {                       // si comptador és múltiple de 10
          System.out.println("");                        // baixa de línia
        }
      }
    }
  }
}
System.out.println("\ns'han mostrat " + comptador + " nombres");
```

exemple 020

Escriu un programa que llig un dígit N entre 1 i 9 de teclat i el nombre de dígits del valor més gran a construir (entre 1 i 18) i calcula N + NN + NNN + NNNN …. visualitza la suma realitzada i el resultat.
Per exemple, per a N = 2 i numDigits = 5 la suma a realitzar és
2 + 22 + 222 + 2222 + 22222.

De la lectura de l'enunciat separa

- les dades
  - un dígit N
  - el nombre de dígits
  - la suma realitzada
  - el resultat
- les accions
  - **llig** un dígit N entre 1 i 9
  - **llig** el nombre de dígits del valor més gran a construir (entre 1 i 18)
  - **calcula** N + NN + NNN + NNNN ….
  - **visualitza** la suma realitzada i el resultat
- les condicions
  - N entre 1 i 9
  - nombre de dígits del valor més gran a construir (entre 1 i 18)

Defineix la informació a manejar

- un dígit, és un **nombre enter** (int), digit
- el nombre de dígits, és **un nombre enter** (int), numDigits
- la suma realitzada, és una cadena de text on es concatena els diferents termes de la suma, no es guarda, es visualitza en la consola
- els diferents termes de la suma, **nombre enter** (long) terme, canvia en cada iteració, el tipus és long, perquè un enter amb 18 dígits no cap en un int.
- el resultat de la suma, és un **nombre enter** (long), suma

Defineix les accions

Llig el valor del dígit, per a obligar l'usuari a introduir un valor entre 1 i 9, el procés de lectura s'escriu en un bucle que es repeteix mentre l'usuari introdueix valors incorrectes, és a dir, mentre (digit < 1 || digit > 9).

Llig el nombre de dígits, per a obligar l'usuari a introduir un valor entre 1 i 18, el procés de lectura s'escriu en un bucle que es repeteix mentre l'usuari introdueix valors incorrectes, és a dir, mentre (numDigits < 1 || numDigits > 18).

suma és l'acumulador dels termes de la suma, és un acumulador de sumes, per tant, s'inicialitza amb 0.

El terme que s'acumula en suma comença amb un dígit i en cada passada se li afig un dígit. terme és un nombre enter i la manera d'afegir un dígit a un nombre en base 10, és multiplicar-lo per 10 (la base) i sumar-li el dígit.

terme = terme * 10 + digit

En el bucle es repeteix la visualització del terme de la suma i el caràcter "+", el canvi del terme i el decrement el comptador de dígits, el bucle es repeteix mentre numDigits és major que zero.

En el bucle la visualització del caràcter "+" està condicionada, no es mostra per a l'últim dígit.

Després del bucle visualitza "=" seguit de la suma.

```
Scanner lector = new Scanner(System.in);
int digit = 0;
do {
   System.out.print("entra un dígit entre 1 i 9 < ");
   digit = lector.nextInt();
} while (digit < 1 || digit > 9);
int numDigits = 0;
do {
   System.out.print("entra el nombre de dígits entre 1 i 18 < ");
   numDigits = lector.nextInt();
} while (numDigits < 1 || numDigits > 18);
long terme = digit;
long suma = 0;
do {
   System.out.print(terme + "");
   suma += terme;
   terme = terme * 10 + digit;
   numDigits--;
   if (numDigits > 0) {
      System.out.print(" + ");
   }
```

```
} while (numDigits > 0);
System.out.println(" = " + suma);
```

Amb mètodes

lligEntre

El mètode permet llegir un enter dins d'un rang:

private static int lligEntre(int inf, int sup, Scanner lector, String missatge)

- rep el valor inferior i el superior, són dos enters inf, sup (paràmetres)
- rep el scanner lector (paràmetre)
- rep el missatge a mostrar missatge (paràmetre)
- torna un enter N que compleix inf <= N <= sup
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- els valors inferior i superior (int), inf i sup
- el scanner, lector
- el missatge a mostrar, missatge
- el valor, és un **nombre enter** (int), num
- Defineix les accions

Rep el scanner, lector, el missatge a mostrar, missatge i dos enters inf i sup, el valor inferior i el valor superior.

Si el valor inferior és major que el valor superior, llavors es crida el mètode amb els valors intercanviats.

Mostra missatge concatenat amb inf i sup.

La lectura es repeteix mentre, el valor llegir està fora del rang definit per inf i sup.

Retorna el valor llegit num.

```
private static int lligEntre(int inf, int sup, Scanner lector, String missatge) {
   if (inf > sup) {
      return lligEntre(sup, inf, lector, missatge);
   }
   int num;
   do {
      System.out.print(missatge + " entre " + inf + " i " + sup + ": ");
      num = lector.nextInt();
   } while (num < inf || num > sup);
   return num;
}
```

main

```
Scanner lector = new Scanner(System.in);
int digit = lligEntre(1, 9, lector, "entra un dígit");
int numDigits = lligEntre(18, 1, lector, "entra el nombre de dígits ");
long terme = digit;
long suma = 0;
do {
   System.out.print(terme + "");
   suma += terme;
   terme = terme * 10 + digit;
   numDigits--;
   if (numDigits > 0) {
      System.out.print(" + ");
   }
} while (numDigits > 0);
System.out.println(" = " + suma);
```

exemple 021

Escriu un programa que llig de teclat el nom i la nòmina de 5 treballadors i visualitza el total de les nòmines, la mitjana de les nòmines i el treballador amb la nòmina més alta i més baixa (nom i nomina).

De la lectura de l'enunciat separa

- les dades
  - el nom de 5 treballadors
  - la nòmina de 5 treballadors
  - el total de les nòmines
  - la mitjana de les nòmines
  - la nòmina més alta
  - la nòmina més baixa
- les accions
  - llig de teclat el nom i la nòmina de 5 treballadors
  - visualitza el total de les nòmines, la mitjana de les nòmines i el treballador amb la nòmina més alta i més baixa
- les condicions
  - la nòmina més alta i més baixa

Defineix la informació a manejar

- el nom d'un treballador, és una **cadena de text** (String), nom
- la nòmina d'un treballador, és **un nombre real** (double), nomina
- el total de les nòmines, és **un nombre real** (double), total
- la mitjana de les nòmines, és **un nombre real** (double), mitjana

- la nòmina més alta, és **un nombre real** (double), maxNomina
- la nòmina més baixa, és **un nombre real** (double), minNomina
- el nom del treballador amb la nòmina més alta, és una **cadena de text** (String), maxNom
- el nom del treballador amb la nòmina més baixa, és una **cadena de text** (String), minNom

Defineix les accions

El total de les nòmines és un acumulador de sumes de nòmines, s'inicialitza amb 0.

Llig de teclat el nom i la nòmina de 5 treballadors, és un bucle que es repeteix 5 vegades, necessitem un comptador, és **un nombre enter** (int) representat per cont.

Suma la nòmina al total de les nòmines.

Les nòmines sol es poden comparar a partir de la segona nòmina, per això la primera lectura és el màxim com el mínim. En la primera lectura el comptador val 1.

Si la nòmina és major que la nòmina major, llavors canvia la nòmina més alta i el nom.

Si la nòmina és menor que la nòmina menor, llavors canvia la nòmina més baixa i el nom.

Repeteix la lectura del nom i nòmina dels treballadors mentre el comptador no arriba a 5.

Calcula la mitjana de les nòmines, la suma de les nòmines dividit per 5.

Visualitza el total de les nòmines, la mitjana de les nòmines i el treballador amb la nòmina més alta i més baixa

```
Scanner lector = new Scanner(System.in);
lector.useLocale(Locale.UK);
System.out.println("Entra les dades dels 5 treballadors");
System.out.print("nom: ");
String nom = lector.nextLine();
System.out.print("nómina: ");
double nomina = lector.nextDouble();          // llig el nom i la nòmina del primer treballador
double total = nomina;                        // la nòmina és el total
double maxNomina = nomina;                    // el primer treballador és el màxim
String maxNom = nom;
double minNomina = nomina;                    // el primer treballador és el mínim
String minNom = nom;
int cont = 1;                                 // ja has tractat el primer treballador
while (cont < 5) {                            // mentre no arriben al 5 treballador
  lector.nextLine();                          // per a buidar el buffer d'entrada
  System.out.print("nom: ");
  nom = lector.nextLine();
```

```
  System.out.print("nómina: ");
  nomina = lector.nextDouble();                  // llig el nom i la nòmina del  treballador
  total += nomina;                               // acumula la nòmina en el total
// si la nòmina és major que el màxim, llavors canvia la nòmina màxima i el nom del treballador
  if (nomina > maxNomina) {
      maxNom = nom;
      maxNomina = nomina;
  }
// si la nòmina és menor que el mínim, llavors  canvien la nòmina mínima i el nom del treballador
  if (nomina < minNomina) {
     minNom = nom;
     minNomina = nomina;
  }
  cont++;                                        // compta el treballador
}
double mitjana = total / 5;                      // calcula la mitjana de les nòmines
System.out.printf("total nómines %.2f\nmitjana de les nómines %.2f\nMàxim: %.2f --> %s\nMínim: %.2f -->
%s",
     total, mitjana, maxNomina, maxNom, minNomina, minNom
);
```

El perquè de la sentència

```
  lector.nextLine();          // per a buidar el buffer d'entrada
```

Quan lector.nextDouble() obté els caràcters del buffer d'entrada per a transformar-los a un double deixa el caràcter retorn de carro en el buffer. El mètode lector.nextLine() obté els caràcters del buffer d'entrada fins al retorn de carro, inclòs. Si no es posa la sentència lector.nextLine(); per a buidar el buffer d'entrada, llavors la sentència nom = lector.nextLine(); sols recupera el retorn de carro deixat per la lectura anterior.

exemple 022

Escriu un programa que llig nombres enters positius i indica si són perfectes, finalitza quan s'introdueix un 0.
Un nombre és perfecte si la suma dels seus divisors excepte ell mateix és igual al nombre.

De la lectura de l'enunciat separa

- les dades
  - nombres enters positius
  - la suma dels seus divisors
- les accions
  - **llig** nombres enters positius
  - **indica** si són perfectes

    - **finalitza** quan s'introdueix un 0
- les condicions
    - llig nombres enters positius
    - finalitza quan s'introdueix un 0
    - és perfecte si la suma dels seus divisors excepte ell mateix és igual al nombre

Defineix la informació a manejar

- un nombre enter, és un **nombre enter** (int), num
- divisor, és **un nombre enter** (int), div
- la suma de divisors, és un **nombre enter** (int), suma

Defineix les accions

Llig un nombre enter de teclat. La lectura del nombre es realitza abans del bucle i al final, d'aquesta forma sempre es troba abans de la comprovació de final del bucle.

Si el nombre és 0 acaba el programa, no comprova si 0 és perfecte o no, és la marca de final del bucle.

Si el nombre no és positiu, llavors es llig el nombre següent.

Un nombre és divisor d'un altre si el mòdul dona 0 (15 % 3 == 0, 3 és divisor de 15).

Has de realitzar un bucle per a trobar els divisors del nombre, des d'1 al nombre (exclusivament) i sumar-los. Visualitza els divisors i la seua suma.

Compara la suma i el nombre llegit, si són iguals, escriu "és perfecte"

Per a cada nombre llegit, has de tornar a inicialitzar la suma dels divisors a 0.

```
Scanner lector = new Scanner(System.in);
System.out.println("Entra nombres enters positius (0 acaba)");
System.out.print("nombre: ");
int num = lector.nextInt();
while (num != 0) {
  if (num > 0) {
    int suma = 0;                          // suma dels divisors
    int div = 1;                           // divisors per al nombre, comença en 1
    System.out.println("divisors de " + num);
    while (div < num) {
      if (num % div == 0) {
        System.out.print(div + "  ");
        suma += div;                       // acumula div en suma
      }
      div++;                               // passa al següent divisor
    }
    System.out.println("\nsuma = " + suma + (suma == num ? " >> el nombre és perfecte" : ""));
  }
```

```
  System.out.print("nombre: ");
  num = lector.nextInt();
}
```

Aquí tens una llista dels primers nombres perfectes: 6, 28, 496, 8128, 33550336, 8589869056, 137438691328, 230584300813995212, ..., si vols comprovar els últims valors, has de canviar el tipus de dades a long i tindre paciència :)

Amb mètodes

esPerfecte

El mètode indica si un nombre és perfecte:

private static boolean esPerfecte(long num) {

- rep el nombre, num (paràmetre)
- torna true si num és perfecte o false si num no és perfecte
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- l'enter representat per num
- el divisor, és un nombre enter (int) amb la referència div
- la suma dels divisors, és un nombre enter (int) amb la referència suma

Defineix les accions

Rep el nombre l'enter num.

El primer divisor és div = 1, la suma dels divisors comença en 0.

Mentre el divisor és menor que num, es comprova si num és divisible per div, si és així el divisor s'acumula en suma.

Retorna el valor de la comparació de suma amb num.

```
private static boolean esPerfecte(long num) {
  long suma = 0;
  long div = 1;
  while (div < num) {
    if (num % div == 0) {
      suma += div;
    }
    div++;
  }
  return suma == num;
}
```

main

```
Scanner lector = new Scanner(System.in);
System.out.println("Entra nombres enters positius (0 acaba)");
System.out.print("nombre: ");
long num = Math.abs(lector.nextLong());
while (num != 0) {
   System.out.println(num+"  "+ (esPerfecte(num) ? " és " : " no és ")+ "perfecte");
   System.out.print("nombre: ");
   num = Math.abs(lector.nextLong());
}
```

exemple 023

Escriu un programa que llig nombres enters positius en base 10 i els visualitza en binari, finalitza quan s'introdueix un 0.

De la lectura de l'enunciat separa

- les dades
  - nombres enters positius en base 10
  - nombres enters positius en base 2
- les accions
  - **llig** nombres enters positius
  - els **visualitza** en binari
  - **finalitza** quan s'introdueix un 0
- les condicions
  - nombres enters positius en base 10
  - finalitza quan s'introdueix un 0

Defineix la informació a manejar

- un nombre enter, és un **nombre enter** (int), numB10
- un nombre binari, és una **cadena de text** (String), numBin, no es fan càlculs amb el nombre binari, és únicament una representació

Defineix les accions

Llig un nombre en base 10 de teclat. La lectura del nombre es realitza abans del bucle i al final, d'aquesta forma sempre es troba abans de la comprovació de final del bucle.

Si el nombre és un 0, acaba el programa. El 0 no es comprova, és la marca de final del bucle.

Si el nombre és menor que 0, llavors es canvia de signe (només es manegen nombres positius).

Has de realitzar un bucle per a obtindre els dígits binaris del nombre, és a dir, els dígits en base 2. Aquests dígits s'obtenen mitjançant el mòdul 2 del nombre.

Un dígit binari, és un nombre enter representat per digit, s'obté fent el mòdul 2 del nombre en base 10

digit = numB10 % 2;

aquest dígit es concatena al text de numBin

numBin = digit + numBin;

El dígit es posa a l'inici del nombre en base 2 que s'està construint.

Aquest dígit que s'ha obtingut del nombre, s'ha d'eliminar del nombre en base 10, per tant, s'ha de dividir per 2

numB10 /= 2;

El bucle de mòduls i divisions es repeteix mentre el nombre en base 10 és major que 0.

```
Scanner lector = new Scanner(System.in);
System.out.println("Entra nombres enters positius (0 acaba)");
System.out.print("nombre: ");
int numB10 = lector.nextInt();
while (numB10 != 0) {
   numB10 = Math.abs(numB10);
   System.out.print(numB10);                  // visualitza el valor decimal del nombre
   String numBin = "";
   while (numB10>0) {                         // mentre el nombre en base 10 és major que 0
      int digit = numB10 % 2;                 // obté el dígit binari
      numBin = digit + numBin;                // concatena el dígit a l'inici del nombre binari
      numB10 /= 2;                            // elimina el dígit tractat
   }
   System.out.print(" = " + numBin + "\n");   // visualitza el valor binari del nombre
   System.out.print("nombre: ");
   numB10 = lector.nextInt();
}
```

Amb mètodes

aBinari

El mètode permet transformar un enter positiu a binari:

private static String aBinari(int num) {

- rep el valor de l'enter a transformar, num (paràmetre)
- torna una cadena de text amb el valor binari
- és privat, sols s'usa en la classe principal

- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- el nombre (int), num
- el dígit binari, és un **nombre enter** (int), digit
- el nombre binari, és una **cadena de text**, numBin

Defineix les accions

Rep l'enter num, el nombre en base 10.

Per a obtindre els dígits binaris del nombre (els dígits en base 2) has de realitzar el mòdul 2 del nombre (torna 0 o 1). El mòdul es guarda en digit que després es concatena al text de numBin. El dígit es posa a l'inici del nombre que s'està construint.

Aquest dígit que s'ha obtingut del nombre, s'ha d'eliminar del nombre en base 10, per tant, s'ha de dividir per 2 (la base).

Has de realitzar un bucle mentre num és major que 0.

Retorna el valor en binari numBin.

```
private static String aBinari(int num) {
   String numBin = "";
   while (num > 0) {
      int digit = num % 2;
      numBin = digit + numBin;
      num /= 2;
   }
   return numBin;
}
```

main

```
Scanner lector = new Scanner(System.in);
System.out.println("Entra nombres enters positius (0 acaba)");
System.out.print("nombre: ");
int numB10 = lector.nextInt();
while (numB10 != 0) {
   numB10 = Math.abs(numB10);
   System.out.print(numB10);
   System.out.print(" = " + aBinari(numB10) + "\n");
   System.out.print("nombre: ");
   numB10 = lector.nextInt();
}
```

exemple 029

Escriu un programa que eleva un nombre n a un exponent ex (n^{ex}), n i ex es lligen de teclat. n i ex són nombres enters. Elevar un nombre a un exponent respon a la fórmula següent

ex = 0 $n^0 = 1$

ex > 0 $n^{ex} = n*n*n*$ *n*n*n hi ha ex productes de n

ex < 0 $n^{ex} = 1 / (n*n*n*$ *n*n*n) hi ha ex productes de n (ex positiu)

La solució es fa amb bucles, usa la funció Math.pow(n,ex) per a comprovar el resultat.

De la lectura de l'enunciat separa

- les dades
  - un nombre
  - un exponent
  - un nombre elevat a un exponent
- les accions
  - **eleva** un nombre n a un exponent
  - n i ex es **lligen** de teclat
  - d'un any **llegit** de teclat
- les condicions
  - ex = 0
  - ex > 0
  - ex < 0

Defineix la informació a manejar

- un nombre n, és **un nombre enter** (int), n
- un exponent ex, és **un nombre enter** (int), ex
- el signe de l'exponent, és **un booleà** (boolean), esNegatiu, val true si l'exponent és menor que zero i false en cas contrari, s'usa per a saber si s'ha de fer la divisió o no
- la potència, és **un nombre real** (double), pot, la divisió té decimals

Defineix les accions

Llig de teclat el nombre n i l'exponent ex.

Dona valor a esNegatiu en funció del valor de ex, si ex és negatiu, li canvia el signe.

Multiplica n, tantes vegades com val ex, és un bucle que es repeteix ex vegades, el resultat s'acumula en pot, és un acumulador de productes, per tant, el valor inicial és 1.

Si esNegatiu és vertader, llavors calcula la inversa, és a dir 1 / pot, sinó ja tenim el resultat en pot.

```
Scanner lector = new Scanner(System.in);
System.out.print("n = ");
int n = lector.nextInt();
```

```
System.out.print("ex = ");
int ex = lector.nextInt();
System.out.println(n + " elevat a " + ex + " és " + Math.pow(n, ex));          // resultat amb Math.pow(n, ex)
boolean esNegatiu = ex < 0;                                                    // guarda si ex és negatiu
ex = Math.abs(ex);                                                             // passa ex a positiu
double pot = 1;                                                                // acumulador de productes, el valor neutre és 1
while (ex > 0) {
   pot *= n;                                                                   // acumula el producte de n
   ex--;
}
if (esNegatiu) {                                                               // si és negatiu, calcula la inversa
   pot = 1 / pot;
}
System.out.println("amb el bucle és " + pot);
```

Amb mètodes

elevatA

El mètode permet elevar un enter a una potència:

private static double elevatA(int num, int ex) {

- rep el nombre i l'exponent, són dos enters, num, ex (paràmetres)
- torna un double amb num elevat a ex
- és privat, sols s'usa en la classe principal
- és estàtic, ja que es va a cridar des del mètode main

Informació a manejar

- un nombre, és un nombre enter (int), num
- un exponent, és un nombre enter (int), ex
- la potència, és un nombre real (double), pot, la divisió té decimals

Defineix les accions

Rep el nombre num i l'exponent ex.

Si l'exponent ex és negatiu, crida al mètode canviant el signe de ex, que torna la inversa, és a dir 1 / pot.

Per a ex positiu, multiplica num, tantes vegades com val ex, és un bucle que es repeteix ex vegades, el resultat s'acumula en pot, és un acumulador de productes, per tant, el valor inicial és 1.

Retorna el valor de pot.

```
private static double elevatA(int num, int ex) {
  if (ex < 0) {
     return 1 / elevatA(num, -ex);
  }
```

```
    double pot = 1;
    while (ex > 0) {
        pot *= num;
        ex--;
    }
    return pot;
}
```

main

```
Scanner lector = new Scanner(System.in);
System.out.print("n = ");
int num = lector.nextInt();
System.out.print("ex = ");
int ex = lector.nextInt();
System.out.println(num + " elevat a " + ex + "\nusant pow     " + Math.pow(num, ex));
System.out.println("usant elevatA " + elevatA(num, ex));
```