

h e l l o

Patricio Pihue
 Andes Matute
 Luis Delfado
 Edmundo Enriguez
 Angie Medina
 Rosaura Espinosa

Clases.

Clase Operaciones matemáticas

- ↳ Suma
- ↳ Resta
- ↳ Multiplicación
- ↳ División.

Clase Electrodomésticos

- ↳ Nevera
- ↳ Cocina
- ↳ Plancha
- ↳ Lavarropas
- ↳ Microondas

Clase marcas de vehículos

- ↳ NISSAN
- ↳ Chevrolet
- ↳ Audi
- ↳ HONDA.

Clase Transportes públicos

- ↳ tren
- ↳ bus
- ↳ taxi
- ↳ metrovías

Clase medios de comunicación

- ↳ Tele
- ↳ Periódico
- ↳ Facebook
- ↳ Twitter

Clase Redes Sociales

- ↳ Facebook
- ↳ Twitter
- ↳ messenger
- ↳ what's up

Clase Operadoras Telefónicas

- ↳ Claro
- ↳ Movistar
- ↳ CNT
- ↳ Twentis

Clases Sistemas Operativos

- ↳ windows
- ↳ Mac
- ↳ Linux
- ↳ Unix.

Clases & Marcas de ropa

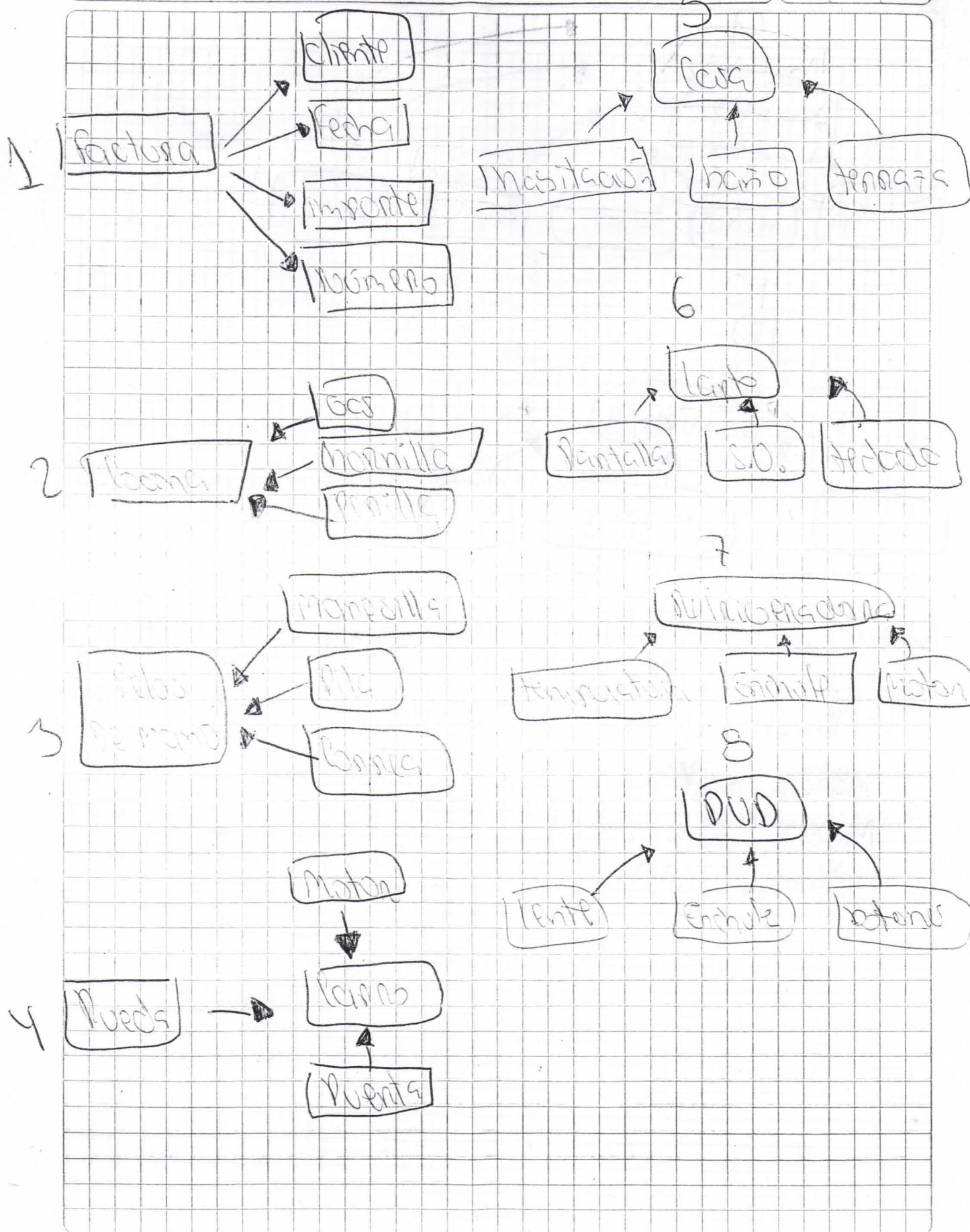
- ↳ Adidas
- ↳ Apollo
- ↳ Lo coste
- ↳ Bershka
- ↳ ZARA.

Clase. Asignaturas

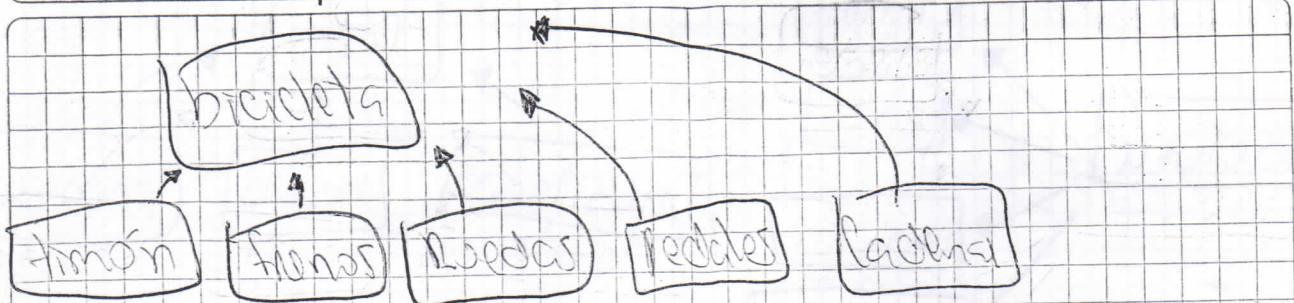
- ↳ lengua
- ↳ matemática
- ↳ sociales
- ↳ música

Comprensión +5

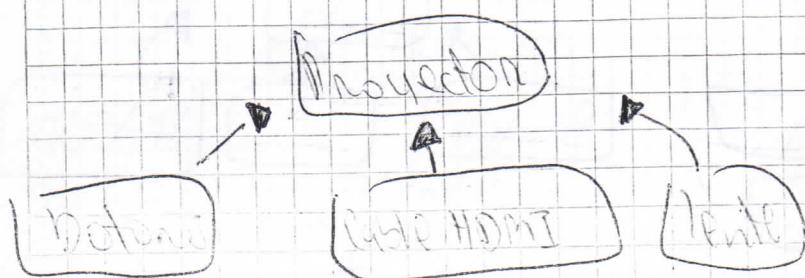
DD MM AA



9



10



Wellington Wong

Edison Gómez

Marcos Wong

Melchor Núñez

100% 71

Herencia

Medina Siguenza
España Angulo
Pinguino
Molte
Enriquez

① Figura Geometrica

Triangulo Cuadro Rectangulo

② AVES

Pato Gallina Ganso

③ Animal

Carnivoro Herbívoro Omnívoro
León Conejo Hombre

④ Transporte

Aereo Terrestre Acuatico
Avion bicicleta bolsa.

⑤ Computadora

Laptop CP Notebook

⑥ Clase Padre

Clase hija1 Clase hija2
Alumno1 Alumno2 Alumno3

⑦ Disciplina

Futbol Basquet Notacion
Atletismo voley

⑧ tipos de Comunicación

Verbal Telefono Correo
Chateos Computadora

⑨ Instrumentos

Guitarra -> Glota

2 Cuerda -> guitarra

1 percusion -> timbal.

⑩ tipos de Claves

bach

optico

teleinfo
qwerty

INTEGRANTES

Pensador, batería, pilas, moneda
cajuelas, moneda tomate MM AA

Objetos

Computadora	Monitor	Microwaves	Lámpara
RPN			Luz
Teclado			Transistor
Mouse			Hacha
			Mesa
Cámera	Flash	Salvavidas	Silla
	Lente	Plata	Plancha
	Botones		Proyector
	Audio		
			Bombín
Coche	Moto		Piedras
	Mando		Palito
	Radio		Tambor
	Perro		
Visiones	Edición de video	Play	Disco
	Cámara	Station	Palanca
	Centro		Moneda
	Útiles de oficina		Lápices
Caja	Motor		Bocina
Acondicionado	Helice	Pantalla	Lámpara
	Manivela		Botones
	Gas		Fuente de luz

soil
water
sunlight
air
minerals

soil
water
sunlight
air
minerals

```
<default config> > > * T D > H > X
```

Q. Search [Ctrl+F]

Projects X Files Services - .../e... B12017_Ej02_28.java X B12017_Ej02_29.java X B22016_Ej02_271.java X B12017_Ej02_30.java X B12017_Ej02_31User.java X B12017_Ej02_015.java X

Source History | Q. * | B12017_Ej02_31User.java | B12017_Ej02_015.java X

Source Packages B12017_Ej02_31Usernar

B12016_Ej03 B12016_Ej02 tes.edu.ec

```
14 public class B12017_Ej02_31Usernar
15 {
16     public static void main(String args[])
17     {
18         public static void main(String args[])
19         {
20             String strUsu, strCsc;
21             try
22             {
23                 InputStreamReader leer=new InputStreamReader(System.in);
24                 BufferedReader leer2=new BufferedReader(ler);
25                 System.out.println("Ingrese su Usuario:");
26                 strUsu=leer2.readLine();
27                 System.out.println("Ingrese su Contraseña:");
28                 strCsc=leer2.readLine();
29                 if(strCsc.equals(strCsc))
30                     System.out.println("Bienvenido " + strUsu);
31                 else
32                     System.out.println("Error la Clave es diferente al de la confirmacion.");
33
34             }
35             catch (IOException e)
36             {
37                 System.out.println("Exception: " + e.toString());
38             }
39             catch (NumberFormatException e)
40             {
41                 System.out.println("Exception: " + e.toString());
42             }
43         }
44     }
45 }
```

```
<default config> <...> B12017_Ej02_28.java X B12017_Ej02_29.java X B22016_Ej02_271.java X B12017_Ej02_30.java X B12017_Ej02_31User.java X B12017_Ej02_015.java X
```

Source History Search (Ctrl+F)

```
13    * @author prof_tes_a
14    */
15    public class B12017_Ej02_015
16    {
17        public static void main(String args [])
18        {
19            String strCada; //cadena de caracteres
20            try
21            {
22                // entrada de datos por teclado InputStreamReader
23                InputStreamReader isr=new InputStreamReader (System.in);
24                BufferedReader bu=new BufferedReader(isr);
25                System.out.print("Ingrese una cadena ");
26                //leo o capturo lo que escribi
27                strCada=bu.readLine();
28                for (int i=0; i<strCada.length();i++)
29                {
30                    System.out.print(strCada.charAt(i)+"-");
31                }
32                }
33                catch (IOException e)
34                {
35                    System.out.println("Exception: " + e.toString());
36                }
37                catch (NumberFormatException e)
38                {
39                    System.out.println("Exception: " + e.toString());
40                }
41                }
42                }
43                }
44                }
45                }
46                }
47                }
48                }
49                }
50                }
51                }
52                }
53                }
54                }
55                }
56                }
57                }
58                }
59                }
60                }
61                }
62                }
63                }
64                }
65                }
66                }
67                }
68                }
69                }
70                }
71                }
72                }
73                }
74                }
75                }
76                }
77                }
78                }
79                }
80                }
81                }
82                }
83                }
84                }
85                }
86                }
87                }
88                }
89                }
90                }
91                }
92                }
93                }
94                }
95                }
96                }
97                }
98                }
99                }
100                }
101                }
102                }
103                }
104                }
105                }
106                }
107                }
108                }
109                }
110                }
111                }
112                }
113                }
114                }
115                }
116                }
117                }
118                }
119                }
120                }
121                }
122                }
123                }
124                }
125                }
126                }
127                }
128                }
129                }
130                }
131                }
132                }
133                }
134                }
135                }
136                }
137                }
138                }
139                }
140                }
141                }
142                }
143                }
144                }
145                }
146                }
147                }
148                }
149                }
150                }
151                }
152                }
153                }
154                }
155                }
156                }
157                }
158                }
159                }
160                }
161                }
162                }
163                }
164                }
165                }
166                }
167                }
168                }
169                }
170                }
171                }
172                }
173                }
174                }
175                }
176                }
177                }
178                }
179                }
180                }
181                }
182                }
183                }
184                }
185                }
186                }
187                }
188                }
189                }
190                }
191                }
192                }
193                }
194                }
195                }
196                }
197                }
198                }
199                }
200                }
201                }
202                }
203                }
204                }
205                }
206                }
207                }
208                }
209                }
210                }
211                }
212                }
213                }
214                }
215                }
216                }
217                }
218                }
219                }
220                }
221                }
222                }
223                }
224                }
225                }
226                }
227                }
228                }
229                }
230                }
231                }
232                }
233                }
234                }
235                }
236                }
237                }
238                }
239                }
240                }
241                }
242                }
243                }
244                }
245                }
246                }
247                }
248                }
249                }
250                }
251                }
252                }
253                }
254                }
255                }
256                }
257                }
258                }
259                }
260                }
261                }
262                }
263                }
264                }
265                }
266                }
267                }
268                }
269                }
270                }
271                }
272                }
273                }
274                }
275                }
276                }
277                }
278                }
279                }
280                }
281                }
282                }
283                }
284                }
285                }
286                }
287                }
288                }
289                }
290                }
291                }
292                }
293                }
294                }
295                }
296                }
297                }
298                }
299                }
300                }
301                }
302                }
303                }
304                }
305                }
306                }
307                }
308                }
309                }
310                }
311                }
312                }
313                }
314                }
315                }
316                }
317                }
318                }
319                }
320                }
321                }
322                }
323                }
324                }
325                }
326                }
327                }
328                }
329                }
330                }
331                }
332                }
333                }
334                }
335                }
336                }
337                }
338                }
339                }
340                }
341                }
342                }
343                }
344                }
345                }
346                }
347                }
348                }
349                }
350                }
351                }
352                }
353                }
354                }
355                }
356                }
357                }
358                }
359                }
360                }
361                }
362                }
363                }
364                }
365                }
366                }
367                }
368                }
369                }
370                }
371                }
372                }
373                }
374                }
375                }
376                }
377                }
378                }
379                }
380                }
381                }
382                }
383                }
384                }
385                }
386                }
387                }
388                }
389                }
390                }
391                }
392                }
393                }
394                }
395                }
396                }
397                }
398                }
399                }
400                }
401                }
402                }
403                }
404                }
405                }
406                }
407                }
408                }
409                }
410                }
411                }
412                }
413                }
414                }
415                }
416                }
417                }
418                }
419                }
420                }
421                }
422                }
423                }
424                }
425                }
426                }
427                }
428                }
429                }
430                }
431                }
432                }
433                }
434                }
435                }
436                }
437                }
438                }
439                }
440                }
441                }
442                }
443                }
444                }
445                }
446                }
447                }
448                }
449                }
450                }
451                }
452                }
453                }
454                }
455                }
456                }
457                }
458                }
459                }
460                }
461                }
462                }
463                }
464                }
465                }
466                }
467                }
468                }
469                }
470                }
471                }
472                }
473                }
474                }
475                }
476                }
477                }
478                }
479                }
480                }
481                }
482                }
483                }
484                }
485                }
486                }
487                }
488                }
489                }
490                }
491                }
492                }
493                }
494                }
495                }
496                }
497                }
498                }
499                }
500                }
501                }
502                }
503                }
504                }
505                }
506                }
507                }
508                }
509                }
510                }
511                }
512                }
513                }
514                }
515                }
516                }
517                }
518                }
519                }
520                }
521                }
522                }
523                }
524                }
525                }
526                }
527                }
528                }
529                }
530                }
531                }
532                }
533                }
534                }
535                }
536                }
537                }
538                }
539                }
540                }
541                }
542                }
543                }
544                }
545                }
546                }
547                }
548                }
549                }
550                }
551                }
552                }
553                }
554                }
555                }
556                }
557                }
558                }
559                }
560                }
561                }
562                }
563                }
564                }
565                }
566                }
567                }
568                }
569                }
570                }
571                }
572                }
573                }
574                }
575                }
576                }
577                }
578                }
579                }
580                }
581                }
582                }
583                }
584                }
585                }
586                }
587                }
588                }
589                }
590                }
591                }
592                }
593                }
594                }
595                }
596                }
597                }
598                }
599                }
600                }
601                }
602                }
603                }
604                }
605                }
606                }
607                }
608                }
609                }
610                }
611                }
612                }
613                }
614                }
615                }
616                }
617                }
618                }
619                }
620                }
621                }
622                }
623                }
624                }
625                }
626                }
627                }
628                }
629                }
630                }
631                }
632                }
633                }
634                }
635                }
636                }
637                }
638                }
639                }
640                }
641                }
642                }
643                }
644                }
645                }
646                }
647                }
648                }
649                }
650                }
651                }
652                }
653                }
654                }
655                }
656                }
657                }
658                }
659                }
660                }
661                }
662                }
663                }
664                }
665                }
666                }
667                }
668                }
669                }
670                }
671                }
672                }
673                }
674                }
675                }
676                }
677                }
678                }
679                }
680                }
681                }
682                }
683                }
684                }
685                }
686                }
687                }
688                }
689                }
690                }
691                }
692                }
693                }
694                }
695                }
696                }
697                }
698                }
699                }
700                }
701                }
702                }
703                }
704                }
705                }
706                }
707                }
708                }
709                }
710                }
711                }
712                }
713                }
714                }
715                }
716                }
717                }
718                }
719                }
720                }
721                }
722                }
723                }
724                }
725                }
726                }
727                }
728                }
729                }
730                }
731                }
732                }
733                }
734                }
735                }
736                }
737                }
738                }
739                }
740                }
741                }
742                }
743                }
744                }
745                }
746                }
747                }
748                }
749                }
750                }
751                }
752                }
753                }
754                }
755                }
756                }
757                }
758                }
759                }
760                }
761                }
762                }
763                }
764                }
765                }
766                }
767                }
768                }
769                }
770                }
771                }
772                }
773                }
774                }
775                }
776                }
777                }
778                }
779                }
780                }
781                }
782                }
783                }
784                }
785                }
786                }
787                }
788                }
789                }
790                }
791                }
792                }
793                }
794                }
795                }
796                }
797                }
798                }
799                }
800                }
801                }
802                }
803                }
804                }
805                }
806                }
807                }
808                }
809                }
810                }
811                }
812                }
813                }
814                }
815                }
816                }
817                }
818                }
819                }
820                }
821                }
822                }
823                }
824                }
825                }
826                }
827                }
828                }
829                }
830                }
831                }
832                }
833                }
834                }
835                }
836                }
837                }
838                }
839                }
840                }
841                }
842                }
843                }
844                }
845                }
846                }
847                }
848                }
849                }
850                }
851                }
852                }
853                }
854                }
855                }
856                }
857                }
858                }
859                }
860                }
861                }
862                }
863                }
864                }
865                }
866                }
867                }
868                }
869                }
870                }
871                }
872                }
873                }
874                }
875                }
876                }
877                }
878                }
879                }
880                }
881                }
882                }
883                }
884                }
885                }
886                }
887                }
888                }
889                }
890                }
891                }
892                }
893                }
894                }
895                }
896                }
897                }
898                }
899                }
900                }
901                }
902                }
903                }
904                }
905                }
906                }
907                }
908                }
909                }
910                }
911                }
912                }
913                }
914                }
915                }
916                }
917                }
918                }
919                }
920                }
921                }
922                }
923                }
924                }
925                }
926                }
927                }
928                }
929                }
930                }
931                }
932                }
933                }
934                }
935                }
936                }
937                }
938                }
939                }
940                }
941                }
942                }
943                }
944                }
945                }
946                }
947                }
948                }
949                }
950                }
951                }
952                }
953                }
954                }
955                }
956                }
957                }
958                }
959                }
960                }
961                }
962                }
963                }
964                }
965                }
966                }
967                }
968                }
969                }
970                }
971                }
972                }
973                }
974                }
975                }
976                }
977                }
978                }
979                }
980                }
981                }
982                }
983                }
984                }
985                }
986                }
987                }
988                }
989                }
990                }
991                }
992                }
993                }
994                }
995                }
996                }
997                }
998                }
999                }
1000                }
1001                }
1002                }
1003                }
1004                }
1005                }
1006                }
1007                }
1008                }
1009                }
1010                }
1011                }
1012                }
1013                }
1014                }
1015                }
1016                }
1017                }
1018                }
1019                }
1020                }
1021                }
1022                }
1023                }
1024                }
1025                }
1026                }
1027                }
1028                }
1029                }
1030                }
1031                }
1032                }
1033                }
1034                }
1035                }
1036                }
1037                }
1038                }
1039                }
1040                }
1041                }
1042                }
1043                }
1044                }
1045                }
1046                }
1047                }
1048                }
1049                }
1050                }
1051                }
1052                }
1053                }
1054                }
1055                }
1056                }
1057                }
1058                }
1059                }
1060                }
1061                }
1062                }
1063                }
1064                }
1065                }
1066                }
1067                }
1068                }
1069                }
1070                }
1071                }
1072                }
1073                }
1074                }
1075                }
1076                }
1077                }
1078                }
1079                }
1080                }
1081                }
1082                }
1083                }
1084                }
1085                }
1086                }
1087                }
1088                }
1089                }
1090                }
1091                }
1092                }
1093                }
1094                }
1095                }
1096                }
1097                }
1098                }
1099                }
1100                }
1101                }
1102                }
1103                }
1104                }
1105                }
1106                }
1107                }
1108                }
1109                }
1110                }
1111                }
1112                }
1113                }
1114                }
1115                }
1116                }
1117                }
1118                }
1119                }
1120                }
1121                }
1122                }
1123                }
1124                }
1125                }
1126                }
1127                }
1128                }
1129                }
1130                }
1131                }
1132                }
1133                }
1134                }
1135                }
1136                }
1137                }
1138                }
1139                }
1140                }
1141                }
1142                }
1143                }
1144                }
1145                }
1146                }
1147                }
1148                }
1149                }
1150                }
1151                }
1152                }
1153                }
1154                }
1155                }
1156                }
1157                }
1158                }
1159                }
1160                }
1161                }
1162                }
1163                }
1164                }
1165                }
1166                }
1167                }
1168                }
1169                }
1170                }
1171                }
1172                }
1173                }
1174                }
1175                }
1176                }
1177                }
1178                }
1179                }
1180                }
1181                }
1182                }
1183                }
1184                }
1185                }
1186                }
1187                }
1188                }
1189                }
1190                }
1191                }
1192                }
1193                }
1194                }
1195                }
1196                }
1197                }
1198                }
1199                }
1200                }
1201                }
1202                }
1203                }
1204                }
1205                }
1206                }
1207                }
1208                }
1209                }
1210                }
1211                }
1212                }
1213                }
1214                }
1215                }
1216                }
1217                }
1218                }
1219                }
1220                }
1221                }
1222                }
1223                }
1224                }
1225                }
1226                }
1227                }
1228                }
1229                }
1230                }
1231                }
1232                }
1233                }
1234                }
1235                }
1236                }
1237                }
1238                }
1239                }
1240                }
1241                }
1242                }
1243                }
1244                }
1245                }
1246                }
1247                }
1248                }
1249                }
1250                }
1251                }
1252                }
1253                }
1254                }
1255                }
1256                }
1257                }
1258                }
1259                }
1260                }
1261                }
1262                }
1263                }
1264                }
1265                }
1266                }
1267                }
1268                }
1269                }
1270                }
1271                }
1272                }
1273                }
1274                }
1275                }
1276                }
1277                }
1278                }
1279                }
1280                }
1281                }
1282                }
1283                }
1284                }
1285                }
1286                }
1287                }
1288                }
1289                }
1290                }
1291                }
1292                }
1293                }
1294                }
1295                }
1296                }
1297                }
1298                }
1299                }
1300                }
1301                }
1302                }
1303                }
1304                }
1305                }
1306                }
1307                }
1308                }
1309                }
1310                }
1311                }
1312                }
1313                }
1314                }
1315                }
1316                }
1317                }
1318                }
1319                }
1320                }
1321                }
1322                }
1323                }
1324                }
1325                }
1326                }
1327                }
1328                }
1329                }
1330                }
1331                }
1332                }
1333                }
1334
```

The screenshot shows the NetBeans IDE interface with the following details:

- Toolbar:** Includes icons for file operations (New, Open, Save, Print, Find, Copy, Paste, Cut, Undo, Redo), a search bar ("Search (Ctrl+F)"), and tabs for "Source", "History", and "Output".
- Project Explorer:** Shows a project named "B22016_Ej02" containing a "Source Packages" node which includes files like B22016_Ej02_01.java through B22016_Ej02_28.java.
- Code Editor:** Displays Java code for a class named B12017_Ej02_28. The code prints a diamond shape to the console using nested loops and string concatenation.

```
1 public static void main(String args [])
2 {
3     String linea=" ";
4     int n=4;
5     int aux=n;
6     for (i=1;i<=n;i++)
7     {
8         //triangulo borde cuadrado
9         for(int j=aux;j>=1;j--)
10        {
11            linea=linea + " ";
12        }
13        System.out.print(linea);
14        linea = " ";
15        aux--;
16    }
17    System.out.println();
18    for(int j=i;j<=k;j++)
19    {
20        linea += " ";
21    }
22    System.out.println(linea);
23    linea = " ";
24 }
25 }
```

- Output Panel:** Shows the command "B22016_Ej02 (run) #2" and the output "Running...".
- Status Bar:** Shows "B22016_Ej02 (run) #2" and the time "29:32".

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

<default config>

Source History ...ave B12017_Ej02_28.java x B12017_Ej02_29.java x B2016_Ej02_271.java x B12017_Ej02_30.java x B12017_Ej02_31User.java x B12017_Ej02_015.java x

Source Services ...ave B12017_Ej02_28.java x B12017_Ej02_29.java x B2016_Ej02_271.java x B12017_Ej02_30.java x B12017_Ej02_31User.java x B12017_Ej02_015.java x

Source Packages

B12016_Ej03

B12016_Ej02

Source Packages

test.adb.ec

```
12    * (author prof. César Á
13    */
14    public class B12017_Ej02_29 {
15        public static void main(String args[]) {
16            String linea = "";
17            int cantidad = 4;
18            int aux = cantidad;
19            int i;
20            for (i = 1; i <= cantidad; i++) {
21                for (int j = aux; j >= i; j--) {
22                    linea += linea + " ";
23                }
24            }
25        }
26    }
27
28    linea = linea + "\n";
29
30    int k = i;
31    k = 2 * k + 1;
32    for (int j = 1; j <= k; j++) {
33        {
34            linea += " ";
35        }
36    }
37    System.out.println(linea);
38    linea = " ";
39    aux--;
40}
41
42}
```

Output

B22016_Ej02 (run) #2

Running... 25:11 INS

(1 more..)

<default config>

Projects Files Services .../src/ B12017_Ej02_28.java X B12017_Ej02_29.java X B22016_Ej02_30.java X B12017_Ej02_31User.java X B12017_Ej02_015.java X 4 ▾

Source History + - ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂ ⌂

Source Packages tes.edu.ec

B12017_Ej02_015.java

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class B12017_Ej02_30 {
    public static void main(String args []) {
        int n=4;
        {
            int i;
            for (i=1;i<n;i++)
            {
                int j;
                for (j=1;j<=n;j++)
                {
                    System.out.print(" * ");
                }
                System.out.print("\n");
            }
        }
    }
}
```

B12017_Ej02_28.java

B12017_Ej02_29.java

B12017_Ej02_30.java

B12017_Ej02_31User.java

B22016_Ej02.java

B22016_Ej02_01.java

B22016_Ej02_02.java

B22016_Ej02_03.java

B22016_Ej02_04.java

B22016_Ej02_05.java

B22016_Ej02_06.java

B22016_Ej02_07.java

B22016_Ej02_08.java

B22016_Ej02_09.java

B22016_Ej02_10.java

B22016_Ej02_11.java

B22016_Ej02_12.java

B22016_Ej02_13.java

B22016_Ej02_14.java

B22016_Ej02_15.java

B22016_Ej02_16.java

B22016_Ej02_17.java

B22016_Ej02_18.java

B22016_Ej02_19.java

B22016_Ej02_20.java

B22016_Ej02_21.java

B22016_Ej02_22.java

B22016_Ej02_23.java

B22016_Ej02_24.java

Non
Organic
Food
Taste
Sweet
Nature
Zen

Tablas de verdad.

Como ya hemos dicho, una fórmula puede tomar los valores verdadero y falso. La semántica es el conjunto de reglas q' permiten dar significado a una fórmula.

Negación. La negación de un valor es su opuesto.

$P \text{ not } P$

0	1
1	0

Conjunción. La conjunción de 2 valores sólo es cierta si ambos son verdaderos.

$P \text{ and } Q$

0 0	0
0 1	0
1 0	0
1 1	1
V V	?

$P \wedge Q$	$P \wedge Q$
V	V
?	?
?	V
?	?

Disyunción. La disyunción de 2 valores sólo es falsa si ambos son falsos.

$P \vee Q$ $P \text{ or } Q$

0 0	1
0 1	1
1 0	0
1 1	1
V V	?

$P \vee Q$	$P \vee Q$
?	?
?	?
?	V
?	?

Disyunción exclusiva. La disyunción exclusiva de 2 valores es verdadera si son diferentes, y falsa si los 2 valores son iguales.

0 0	0
0 1	1
1 0	0
1 1	1
V V	?

Tablas de verdad.

Valor de verdad. Las tablas de verdad son un elemento de la lógica proposicional para terminal el valor de verdad (es decir, si es "verdadero" o "falso") de una proposición.

Proposición. Una proposición es una afirmación capaz de tener un valor de verdad.

Ejemplo 1.

"El día está soleado" (será verdadero o falso, según si este o no

está soleado).

Operaciones lógicas.

una operación lógica se compone de operandos (propósito)

operadores.

Mediante una operación lógica se unen proposición para obtener una nueva proposición compuesta.

tabla de verdad de cada operador lógico.

Conjunción

P	q	$P \wedge q$
V	V	V
V	F	F
F	V	F
F	F	F

Disyunción

P	q	$P \vee q$	$P \vee q$
V	V	V	V
V	F	V	V
F	V	V	V
F	F	F	F

Negación

P	$\neg P$
V	F
F	V

Implicación

P	q	$P \Rightarrow q$	$P \Rightarrow q$
V	V	V	V
V	F	F	F
F	V	V	V
F	F	V	V

Doble implicación

P	q	$P \Leftrightarrow q$
V	V	V
V	F	F
F	V	F
F	F	V

Operadores booleanos

Nombre	Operador	uf. Proceder	Resultado
AND	&&	$A \& \& B$	verdaderas cuando A y B son verdaderas.
OR		$A B$	verdaderas cuando A o B son verdaderos.
NOT	!	$!A$	verdaderas si A es falso.
OR.		$A B$	verdaderas cuando A + B
XOR	^	$A ^ B$	son verdaderos, siempre cuando ambos operandos,
			verdaderas cuando A + B son diferentes.

MARCOS MORA

GENERAR NUMEROS EN JAVA

Podemos **generar números aleatorios** en Java de dos formas distintas:

1. Utilizando el método estático random de la clase Math: Math.random()
2. Utilizando la clase Random

1. Generar números aleatorios utilizando Math.random()

El método random() de la clase Math devuelve un número al azar positivo de tipo double mayor o igual que 0.0 y menor que 1.0

Por ejemplo, el siguiente for genera 5 números aleatorios

```
for(int i = 1; i<=5; i++)
    System.out.println(Math.random());
```

genera 5 números aleatorios que podrían ser estos:

```
0.6586423340678433
0.35474701449674206
0.9552201267900652
0.8309552833908893
0.05210677512170114
```

Para obtener un número entero entre 1 y un límite N, hay que multiplicar el número aleatorio obtenido por N, sumarle 1 y convertirlo a entero:

```
(int)(Math.random()*N + 1);
```

Por ejemplo, para obtener 5 números aleatorios enteros entre 1 y 6:

```
for(int i = 1; i<=5; i++)
    System.out.println((int)(Math.random()*6 + 1));
```

Para obtener un número entero entre dos valores DESDE , HASTA, ambos incluidos, debemos usar la fórmula:

```
(int)(Math.random()*(HASTA-DESDE+1)+DESDE);
```

Por ejemplo, para generar 5 números enteros al azar entre 8 y 15:

```
for(int i = 1; i<=5; i++)
    System.out.println((int)(Math.random()*(15-8+1)+8));
```

2. Generar números aleatorios utilizando la clase Random

La clase Random proporciona un generador de números aleatorios más flexible que el método random de la clase Math anterior.

Se encuentra en el paquete java.util.

Para obtener un número aleatorio utilizando la clase Random debes seguir estos pasos:

1. Importar la clase: import java.util.Random;
2. Crear un objeto de la clase Random
3. Utilizar uno de los métodos de la clase para obtener el número

Algunos métodos de la clase Random:

nextInt() devuelve un número entero positivo o negativo dentro del rango de enteros.

nextInt(int n) devuelve un número entero ≥ 0 y menor que n.

nextDouble() Devuelve un número positivo de tipo double mayor o igual que 0.0 y menor que 1.0

Por ejemplo, para generar 5 enteros al azar:

```
Random rnd = new Random();
for(int i = 1; i<=5; i++)
    System.out.println(rnd.nextInt());
```

Un posible resultado de este for es:

```
-394395199
1133763686
-424454120
1147379979
-2049113297
```

Para generar 5 enteros entre 0 y 6:

```
for(int i = 1; i<=5; i++)
    System.out.println(rnd.nextInt(7));
```

Para generar 5 enteros entre 1 y 6:

```
for(int i = 1; i<=5; i++)
    System.out.println(rnd.nextInt(6)+1);
```

En general, para generar enteros al azar entre dos límites DESDE , HASTA, ambos incluidos:
rnd.nextInt(HASTA-DESDE+1)+DESDE

Por ejemplo, para generar 5 números aleatorios entre 10 y 20:

```
for(int i = 1; i<=5; i++)
    System.out.println(rnd.nextInt(20-10+1)+10);
```



LICENCIAMIENTO DE SOFTWARE

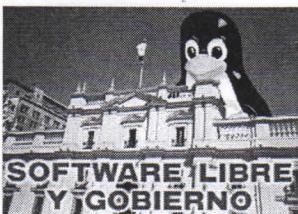
SOFTWARE COMERCIAL

El Software comercial es el **software desarrollado por una empresa con el objetivo de lucrar con su utilización**. Nótese que "comercial" y "proprietario" no son lo mismo. La mayor parte del software comercial es propietario, pero existe software libre que es comercial, y existe software no-libre que no es comercial.

Licencia de software de libre no protegido con copyleft

El software libre no protegido con copyleft viene desde el autor con autorización para redistribuir y modificar así como para añadirle restricciones adicionales. Si un programa es libre pero no protegido con copyleft, entonces algunas copias o versiones modificadas pueden no ser libres completamente. Una compañía de software puede compilar el programa, con o sin modificaciones, y distribuir el archivo ejecutable como un producto privativo de software.

El Sistema X Windows ilustra esto. El Consorcio X libera X11 con términos de distribución que lo hacen software libre no protegido con copyleft.



SOFTWARE PROTEGIDO CON COPYLEFT

El término Copyleft se puede interpretar como Copia permitida, en contraposición a Copyright, o Copia reservada (derechos de autor). En el tema que nos ocupa, se refiere a la autorización por parte del propietario de la licencia para su copia, modificación y posterior distribución, contrariamente a lo que ocurre con el software licenciado bajo los términos de los derechos de autor.

Reclutos
- Patrick Naughton Ing.
Reduta
- James Gosling
Mike Sheridan

Historia del lenguaje Java



Java nació por los 90.

- Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principio de los años 90's.

En Diciembre de **1990** Patrick Naughton, ingeniero de **Sun Microsystems**, reclutó a varios colegas entre ellos James Gosling y Mike Sheridan para trabajar sobre un nuevo proyecto conocido como "*El proyecto verde*".

Con la ayuda de otros ingenieros, empezaron a trabajar en una pequeña oficina en Sand Hill Road en Menlo Park, California. Y así interrumpió todas las comunicaciones regulares con Sun y trabajó sin descanso durante 18 meses.

Intentaban desarrollar una nueva tecnología para programar la siguiente generación de dispositivos inteligentes, en los que Sun veía un campo nuevo a explorar. Crear un lenguaje de programación fácil de aprender y de usar.

En un principio se consideraba C++ como lenguaje a utilizar, pero tanto Gosling como Bill Joy lo encontraron inadecuado. Gosling intentó primero extender y modificar C++ resultando el lenguaje C++ ++ - (++ - porque se añadían y eliminaban características a C++), pero lo abandonó para crear un nuevo lenguaje desde cero al que llamo **Oak** (roble en inglés, según la versión mas aceptada, por el roble que veía através de la ventana de su despacho).

El resultado fue un lenguaje que tenía similitudes con C, C++ y **Objetive C** y que no estaba ligado a un tipo de CPU concreta.

Mas tarde, se cambiaría el nombre de **Oak a Java**, por cuestiones de propiedad intelectual, al existir ya un lenguaje con el nombre de Oak. Se supone que le pusieron ese nombre mientras tomaban café (Java es nombre de un tipo de café, originario de Asia), aunque otros afirman que el nombre deriva de las siglas de James Gosling, Arthur Van Hoff, y Andy Bechtolsheim.

En **Agosto de 1991** Oak ya corría sus primeros programas.

Para **1992**, el equipo ya había desarrollado un sistema en un prototipo llamado **Star7** (*7), dispositivo parecido a una PDA, cuyo nombre venía de la combinación de teclas del teléfono de la oficina del Proyecto Green que permitía a los usuarios responder al teléfono desde cualquier lugar.

Por su parte, el presidente de la compañía Sun, **Scott McNealy**, se dio cuenta de forma oportuna y estableció el **Proyecto Verde** como una subsidiaria de Sun.

Después de mostrar a Scott McNealy y Bill Joy los prototipos de bajo nivel del sistema, continuán con el desarrollo, incluyendo sistema operativo, Green OS; el lenguaje Oak, las librerías, alguna aplicación básica y el hardware, hasta que el **3 de septiembre de 1992** se termina el desarrollo y con ello el **Proyecto Verde**.



se
a
m
o
n
t
e
c
e

JOY

JavaDoc

Proyecto
Sistemas Informáticos
Tecnológico Espíritu Santo

Marcos Mora & Patricio Pihuave
Mayo 2017

JavaDoc

Proyecto
Sistemas Informáticos
Tecnológico Espíritu Santo

Marcos Mora & Patricio Pihuave
Mayo 2017

Dedicatoria

Dedicado a Dios por darme las fuerzas para continuar en lo adverso, por guiarme con sabiduría en las situaciones difíciles. A mis padres por darme la vida, la oportunidad de estudiar y luchar día a día para que lograra escalar un peldaño más.

Y a mis amistades quienes me han demostrado su apoyo incondicional y sincero y a los maestros por sus conocimientos impartidos y guía.

Agradecimientos

A Dios por darme el aliento de vida y ayudarme a superar el miedo para poder continuar, a mis padres por darme el aliento a diario y a mis amigos por estar en las buenas y malas situaciones.

A mi maestra por su paciencia para conmigo, así como por sus conocimientos impartidos en clases ya que me ayudaron a poder pensar antes de cualquier situación a hacer una buena elección.

Tabla de Contenidos

Capítulo 1 Introducción e información general	5
Tipos de comentarios en java	¡Error! Marcador no definido.
Etiquetas de java	¡Error! Marcador no definido.
Donde utilizar javadoc	¡Error! Marcador no definido.
Ejemplos de internet	¡Error! Marcador no definido.
Conclusion	24
Referencias	¡Error! Marcador no definido.

Capítulo 1

Java

Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible.

JavaDoc

Javadoc es una utilidad de Oracle para la generación de documentación de APIs Application Programming Interface (Interfaz de programación de aplicaciones) en formato HTML a partir de código fuente Java. Javadoc es el estándar para documentar clases de Java. La mayoría de los IDEs Integrated Drive Electronics (entorno de desarrollo integrado) utilizan javadoc para generar de forma automática documentación de clases.

Javadoc es la herramienta de java para generar documentación básica para el programador a partir del código fuente. Se intenta evitar que la documentación que se genera mediante un editor de texto se quede rápidamente obsoleta cuando el programa continúa su desarrollo y no se tiene la disciplina/tiempo para mantener la documentación al día.

Una necesidad que sólo se aprecia en su debida magnitud cuando hay errores que reparar o hay que extender el programa con nuevas capacidades o adaptarlo a un nuevo escenario. Hay dos reglas que no se deben olvidar nunca:

1. todos los programas tienen errores y descubrirlos sólo es cuestión de tiempo y de que el programa tenga éxito y se utilice frecuentemente

2. todos los programas sufren modificaciones a lo largo de su vida, al menos todos aquellos que tienen éxito

Por una u otra razón, todo programa que tenga éxito será modificado en el futuro, bien por el programador original, bien por otro programador que le sustituya. Pensando en esta revisión de código es por lo que es importante que el programa se entienda: para poder repararlo y modificarlo.

Documentar un proyecto es algo fundamental de cara a su futuro mantenimiento. Cuando programamos una clase, debemos generar documentación lo suficientemente detallada sobre ella como para que otros programadores sean capaces de usarla sólo con su interfaz. No debe existir necesidad de leer o estudiar su implementación, lo mismo que nosotros para usar una clase del API (Application Programming Interface - Interfaz de programación de aplicaciones) Java no leemos ni estudiamos su código fuente.

Tipos de Comentarios en Java

Java admite tres tipos de comentarios. Los dos primeros son el lineal que se representa con “//” y el multilinea “/* */”.

El tercer tipo se llama Javadoc o comentario de documentacion, comienzan con los caracteres “/**”, se pueden prolongar a lo largo de varias líneas (que probablemente comiencen con el carácter “*”) y terminan con los caracteres “*/”. Los comentarios de la documentación le permiten integrar información sobre su programa en el propio programa. A continuación, puede utilizar el programa de utilidad javadoc (suministrado con el JDK) para extraer la información y ponerla en un archivo HTML.

Etiquetas de Javadoc

La utilidad javadoc reconoce las siguientes etiquetas:

@author

La etiqueta @author documenta al autor de una clase o interfaz, tiene la siguiente sintaxis:

@author descripción

Aquí, la descripción ejecutando javadoc para que el campo @author se incluya en la documentación HTML.

@Version

La etiqueta de versión especifica la versión de una clase o interfaz. Tiene la siguiente sintaxis:

@Version info

Aquí, info es una cadena que contiene información de versión, normalmente una versión número, como 2.2. Deberá especificar la opción de versión al ejecutar javadoc para que el campo de versión se incluya en la documentación HTML.

@param

La etiqueta @param documenta un parámetro. Tiene la siguiente sintaxis:

`@param parameter-name explanation`

Aquí, parameter-name especifica el nombre de un parámetro. El significado de ese parámetro se describe mediante una explicación. La etiqueta @param se puede utilizar sólo en la documentación de un método o constructor, o una clase genérica o interfaz.

@return

La etiqueta de @return describe el valor devuelto de un método. Tiene la siguiente sintaxis:

`@return explicacion`

Aquí, la explicación describe el tipo y el significado del valor devuelto por un método. La etiqueta de @return (retorno) sólo se puede utilizar en la documentación de un método.

@see

La etiqueta @see Proporciona una referencia a información adicional. Dos formas de uso común se muestran aquí:

`@see anchor`

@see pkg.class#member text

En la primera forma ancla es un enlace a una URL absoluta o relativa. En la segunda forma ver `pkg.Class#member` texto especifica el nombre del elemento y el texto es el texto que se muestra para ese elemento. El parámetro de texto es opcional y, si no se utiliza, se muestra el elemento especificado por `pkg.Class#member`. El nombre de miembro también es opcional. Por lo tanto, puede especificar una referencia a un paquete, clase o interfaz además de una referencia a un método o campo específico. El nombre puede ser totalmente calificado o parcialmente calificado.

@Thows

La etiqueta `@Throws` (lanza) tiene el mismo significado que la etiqueta de excepción.

@deprecated

La etiqueta `@deprecated` especifica que los elementos de un programa están obsoletos. Se recomienda incluir etiquetas `@see` o `{@link}` para informar al programador sobre las alternativas disponibles. La sintaxis es la siguiente:

@deprecated description

Aquí, la descripción es el mensaje que describe la depreciación. La etiqueta `@deprecated` se puede utilizar en la documentación de campos, métodos, constructores, clases e interfaces.

@exception

La etiqueta `@exception` describe una excepción, describe una excepción a un método. Tiene la siguiente sintaxis:

@exception exception-name explanation

Aquí, el nombre completo de la excepción se especifica por el nombre de la excepción, y la explicación es una cadena que describe cómo puede producirse la excepción. La etiqueta @exception sólo se puede utilizar en la documentación de un método o constructor.

@Since

La etiqueta @Since establece que un elemento fue introducido en una versión específica. Tiene la siguiente sintaxis:

@Since release

Aquí, release es una cadena que designa el lanzamiento o versión en la que esta característica se hizo disponible.

@Value

La etiqueta @Value tiene dos formas, la primera muestra el valor de la constante que precede, que debe ser un campo estático.

@serial

La etiqueta @serial define el comentario para un campo serializable predeterminado. Tiene la siguiente sintaxis:

@serial descripción

Aquí, la descripción es el comentario para ese campo.

@serialData (serie de datos)

La etiqueta @serialData documenta los datos escritos por los métodos Escribir Objeto () y escribir Métodos externos. (). Tiene la siguiente sintaxis:

@SerialData descripción

Aquí, la descripción es el comentario para esos datos.

@code

La etiqueta **@code** le permite incrustar texto, como un fragmento de código, en un comentario. Ese texto se muestra como-es en la fuente de código, sin ningún procesamiento posterior, como la representación HTML.

{@docRoot}

Especifica la ruta al directorio raíz de la documentación actual.

{@inheritDoc}

Esta etiqueta hereda un comentario de la superclase inmediata

¿Dónde utilizar javadoc?

1. Documentación de clases e interfaces

Deben usarse al menos las etiquetas:

@author: nombre del autor.

@version: identificación de la versión y fecha.

@see: referencia a otras clases y métodos

2. Documentación de constructores y métodos

Deben usarse al menos las etiquetas:

@param: una por argumento de entrada

@return: si el método no es void

@exception o @throws:

Una por tipo de Exception que se puede lanzar (@exception y @throws se pueden usar indistintamente).

Las etiquetas @author y @version se usan para documentar clases e interfaces.

Por tanto no son válidas en cabecera de constructores ni métodos. La etiqueta @param se usa para documentar constructores y métodos. La etiqueta @return se usa solo en métodos de tipo función.

Dentro de los comentarios se admiten etiquetas HTML, por ejemplo con @see se puede referenciar una página web como link para recomendar su visita de cara a ampliar información.

1. Para indicarle a Javadoc que queremos incluir documentación, debemos comenzar los comentarios de la siguiente manera:

```
/***
 * Esto para Javadoc
 */
```

2. Creamos una clase llamada "Clase_java" y en el encabezado colocamos el autor de la clase, la instrucción @see con un enlace a una página web, también podemos agregar la versión de la clase:

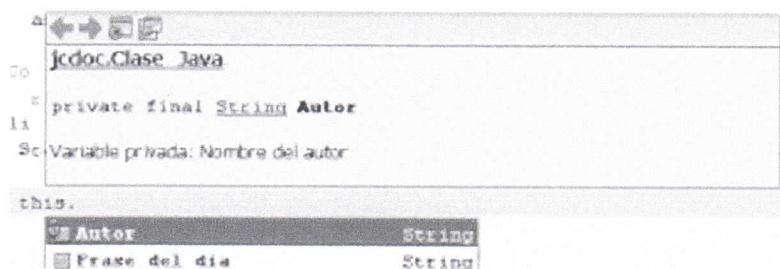
```
/**
 * @author Patricio
 * @see <a href="http://www.google.com">http://www.google.com</a>
 * @version 1.2 17 de Mayo de 2017
 */
public class Clase_Java {
}
```

3. También podemos documentar las variables que hacen parte de nuestro código.

```
/**
 * Variable privada: Nombre del autor
 */
private final String Autor = "Patricio";

/**
 * Variable publica: una frase para reflexionar
 */
public String Frase_del_dia = "Carpe diem";
```

Cuando se haga uso de estas variables comentadas, en el editor de Netbeans podremos ver algo como esto:



4. No debemos olvidar también comentar el constructor de clase.

```
/**
 * Constructor de clase
```

```
/*
public clase_Java(){
}
```

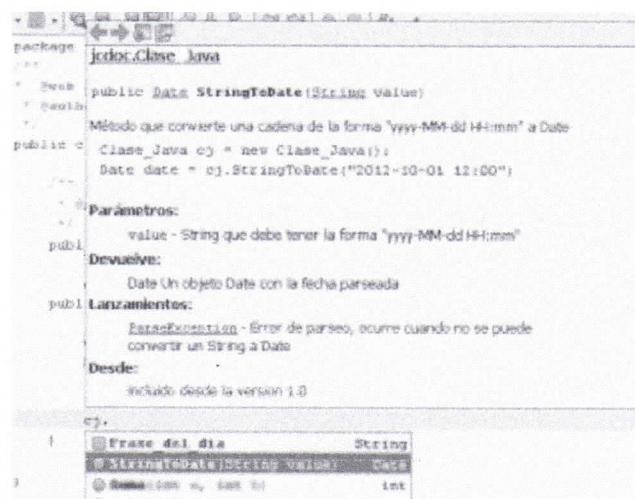
Ejemplos del Internet

Primeros

Cuando creamos un método es recomendable documentar los parámetros de entrada, si es que los tuviera, y si este método retorna algún resultado, utilizar la marca @return, si por el contrario el método es de la forma VOID, no se usa nada. Así también para las excepciones que puedan ocurrir se usa @exception. En la descripción del método, se puede incluir un ejemplo de uso encerrado en las etiquetas PRE, por ejemplo:

```
/**
 * Método que convierte una cadena de la forma "yyyy-MM-dd HH:mm" a
Date
 * <PRE> Clase_Java cj = new Clase_Java();
 * Date date = cj.StringToDate("2012-10-01 12:00")</PRE>
 * @param value String que debe tener la forma "yyyy-MM-dd HH:mm"
 * @return Date Un objeto Date con la fecha parseada
 * @exception ParseException Error de parseo, ocurre cuando no se
puede convertir un String a Date
 * @since incluido desde la versión 1.0
 */
public Date StringToDate( String value )
{
    Date date = new Date();
    SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd
HH:mm");
    try {
        date = (java.util.Date) formatter.parse( value );
    } catch (ParseException ex) {
        System.err.println( ex.getMessage() );
    }
    return date;
}
```

Cuando utilizamos este método, podremos ver que Netbeans nos despliega toda la información en cuando se hace referencia al nombre StringToDate.



Segundo

Primer ejemplo: Utiliza comentarios de documentación A continuación se muestra un programa de ejemplo que utiliza la documentación comments. Notice la forma en que cada comentario precede inmediatamente el elemento que describe. Después de ser procesado por javadoc, la documentación sobre la clase SquareNum se encontrará en SquareNum.html.

```
/*
 * To change this license header, choose License Headers in Project
Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
import java.io.*;
/**
 * Esta clase demuestra comentarios de documentación
 * @author Patricio
 * @version 1.2
 */
public class SquareNum {
/**
 * Este método devuelve el cuadrado de numero.
 * Esta es una descripción multilínea. Puedes usar
 * Tantas líneas como quieras.
 * @param num el valor a ser cuadrado.
 * @return num squared.
 */
public double square(double num){
    return num * num;
}
/**
 * Este método introduce un número del usuario.
 * @return la entrada de valor como un doble.
 * @exception IOException En error de entrada.
 * @see IOException
 */
public double getNum() throws IOException {
    // Crear un lector de Buffered usando el System.in
    InputStreamReader isr = new InputStreamReader(System.in);
    BufferedReader inData = new BufferedReader (isr);
    String str;

    str = inData.readLine();
    return (new Double(str)).doubleValue();
}
```

```
}

/**
 * Este método demuestra cuadrado ().
 * @param args no utilizado.
 * @exception IOException en error de entrada.
 * @see IOException
 */
public static void main (String args[])
    throws IOException
{
    SquareNum ob = new SquareNum();
    double val;
    System.out.println("Introduzca el valor a cuadrar:");
    val = ob.getNum ();
    val = ob.square (val);
    System.out.println("El valor cuadrado es" + val);
}
}
```

Javadoc

Class SquareNum

java.lang.Object
SquareNum

```
public class SquareNum
extends java.lang.Object
```

Esta clase demuestra comentarios de documentación

Constructor Summary

Constructors

Constructor and Description

SquareNum()

Method Summary

All Methods

Static Methods

Instance Methods

Concrete Methods

Modifier and Type

Method and Description

double	getNum()	Este método introduce un número del usuario.
static void	main(java.lang.String[] args)	Este método demuestra cuadrado () .
double	square(double num)	Este método devuelve el cuadrado de numero.

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

SquareNum

```
public SquareNum()
```

Method Detail**square**

```
public double square(double num)
```

Este método devuelve el cuadrado de numero. Esta es una descripción multilínea. Puedes usar Tantas líneas como quieras.

Parameters:

num - el valor a ser cuadrado.

Returns:

num squared.

getNum

```
public double getNum()  
    throws java.io.IOException
```

Este método introduce un número del usuario.

Returns:

la entrada de valor como un doble.

Throws:

java.io.IOException - En error de entrada.

See Also:

[IOException](#)

main

```
public static void main(java.lang.String[] args)  
    throws java.io.IOException
```

Este método demuestra cuadrado () .

Parameters:

args - no utilizado.

Throws:

java.io.IOException - en error de entrada.

See Also:

[IOException](#)

Ejemplos nuestros

Primero Patricio

```

import java.util.ArrayList;
import java.util.Random;

/**
 * Esta clase define objetos que contienen tantos enteros aleatorios entre 0 y
1000 como se le definen al crear un objeto.
 * @author Patricio
 * @version 1.0
 *
 */
public final class SerieDeAleatoriosD
{
    //Campos de la clase
    private final ArrayList<Integer> serieAleatoria;
    /**
     * Constructor para la serie de números aleatorios
     * @param numeroItems El parámetro numeroItems define el número de
elementos que va a tener la serie aleatoria
     */
    public SerieDeAleatoriosD (int numeroItems) {
        serieAleatoria = new ArrayList<> ();
        for (int i=0; i<numeroItems; i++) { serieAleatoria.add(0); }
        System.out.println ("Serie inicializada. El número de elementos en la
serie es: " + getNumeroItems() );
    } //Cierre del constructor
    /**
     * Método que devuelve el número de ítems (números aleatorios) existentes
en la serie
     * @return El número de ítems (números aleatorios) de que consta la serie
     */
    public int getNumeroItems() { return serieAleatoria.size(); }
    /**
     * Método que genera la serie de números aleatorios
     */
    public void generarSerieDeAleatorios () {
        Random numAleatorio;
        numAleatorio = new Random ();
        for (int i=0; i < serieAleatoria.size(); i++) { serieAleatoria.set(i,
numAleatorio.nextInt(1000) ); }
        System.out.print ("Serie generada! ");
    } //Cierre del método
} //Cierre de la clase y del ejemplo

```

JavaDoc

Class SerieDeAleatoriosD

java.lang.Object
 SerieDeAleatoriosD

```
public final class SerieDeAleatoriosD
extends java.lang.Object
```

Esta clase define objetos que contienen tantos enteros aleatorios entre 0 y 1000 como se le definen al crear un objeto.

Constructor Summary

Constructors

Constructor and Description

SerieDeAleatoriosD(int numeroItems)

Constructor para la serie de números aleatorios

Method Summary

All Methods

Instance Methods

Concrete Methods

Modifier and Type

Method and Description

void

generarSerieDeAleatorios()

Método que genera la serie de números aleatorios

int

getNumeroItems()

Método que devuelve el número de ítems (números aleatorios) existentes en la serie

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail**SerieDeAleatoriosD**

```
public SerieDeAleatoriosD(int numeroItems)
```

Constructor para la serie de números aleatorios

Parameters:

numeroItems - El parámetro numeroItems define el número de elementos que va a tener la serie aleatoria

Method Detail**getNumeroItems**

```
public int getNumeroItems()
```

Método que devuelve el número de ítems (números aleatorios) existentes en la serie

Returns:

El número de ítems (números aleatorios) de que consta la serie

generarSerieDeAleatorios

```
public void generarSerieDeAleatorios()
```

Método que genera la serie de números aleatorios

Ejercicio Marcos

package figuras;

```
/*
 * Una clase para representar círculos situados sobre el plano.
 * Cada círculo queda determinado por su radio junto con las
 * coordenadas de su centro.
 * @version 1.2, 24/12/04
 * @author Rafa Caballero
 */

public class Círculo {
    protected double x; // coordenadas del centro
    protected double r; // radio del círculo

    /**
     * Crea un círculo a partir de su origen su radio.
     * @param x La coordenada x del centro del círculo.
     * @param y La coordenada y del centro del círculo.
     * @param r El radio del círculo. Debe ser mayor o igual a 0.
     */
    public Círculo(double x, double y, double r) {
        this.x = x; this.y = y; this.r = r;
    }

    /**
     * Cálculo del área de este círculo.
     * @return El área (mayor o igual que 0) del círculo.
     */
    public double área() {
        return Math.PI * r * r;
    }

    /**
     * Indica si un punto está dentro del círculo.
     * @param px componente x del punto
     * @param py componente y del punto
     * @return true si el punto está dentro del círculo o false en otro caso.
     */
    public boolean contiene(double px, double py) {
        /* Calculamos la distancia de (px,py) al centro del círculo (x,y),
         * que se obtiene como raíz cuadrada de (px-x)^2+(py-y)^2 */
        double d = Math.sqrt((px - x) * (px - x) + (py - y) * (py - y));

        // El círculo contiene el punto si d es menor o igual al radio
        return d <= r;
    }
}
```

Conclusión

Habiéndose dado argumentos que el Javadoc realmente es útil y necesario para documentar el código para nosotros mismo en futuras revisiones y/o modificaciones y como para los demás desarrolladores, es importante recalcar que todo desarrollador debería como regla general incluir en sus proyectos en java los comentarios de documentación.

Lista de referencias

Java concepto tomado de A Beginner's Guide Sixth Edition Create, Compile, and Run Java Programs Today, Autor Herbert Schildt. Part V Applying Java.

Ejemplos de internet, ejemplo 1 tomado de:

<http://jc-mouse.blogspot.com/2012/07/creacion-y-uso-de-javadoc-con-netbeans.html>

Ejemplo 2 tomado de A Beginner's Guide Sixth Edition Create, Compile, and Run Java Programs Today, Autor Herbert Schildt. Part V Applying Java.

Ejemplos nuestros, ejemplo 1 Patricio Pihuave.