

```

1  /*for loop: for loop is the best choice if we know the number of iterations in advance.
2  It is also called entry controlled loop.
3  Syntax:
4  for(initialization;condition;increment/decrement)
5  {
6      Statement or code to be executed;
7  }
8  */
9  /*-----*/
10
11 /*#1WAP to print numbers from 1 to 10 using for loop.*/
12 import java.util.*;
13 class Test1
14 {
15     public static void main(String[] args)
16     {
17         for(int i=1;i<=10;i++)
18         {
19             System.out.print(i+" ");
20         }
21     }
22 }
23 /* OUTPUT
24 1 2 3 4 5 6 7 8 9 10
25 */
26 /*-----*/
27
28 /* #2 Curly Braces are optional. we can take only one statement under for loop
29 without curly braces. */
30 //import java.util.*;
31 class Test2
32 {
33     public static void main(String[] args)
34     {
35         for(int i=1;i<=10;i++)
36
37             System.out.print(i+" ");
38     }
39 }
40 /* OUTPUT
41 1 2 3 4 5 6 7 8 9 10
42 */
43 /*-----*/
44
45 /* #3 Curly Braces are optional. we can take only one statement under for loop
46 without curly braces which should not be declarative statement. */
47 //import java.util.*;
48 class Test3
49 {
50     public static void main(String[] args)
51     {
52         //for(int i=1;i<=10;i++)
53
54             //int x;
55
56     }
57 }
58 /* OUTPUT
59 L1.java:50: error: variable declaration not allowed here
60
61
62
63 1 error
64 */
65 /*-----*/
66
67 /* #4 We can take declarative statement under for loop using curly Braces. */
68 //import java.util.*;
69 class Test4

```

```

70 {
71     public static void main(String[] args)
72     {
73         for(int i=1;i<=10;i++)
74         {
75             int x=10;
76         }
77     }
78 }
79 /* OUTPUT
80 No compiler error
81 */
82 /*
83 -----
84 /* #5 Initialization section will be executed only once. Here we declare
85 and initialize local variables for for loop.
86 int i=0,j=0; can declare multiple variables but should be of same type.
87 int i=0,String s="Amit"; it is not valid because we can declare any number of variables
88 but
89 should be of same type.
90 int i=0,int j=0; it is also not valid. We need to write data type only once.*/
91 //import java.util.*;
92 class Test5
93 {
94     public static void main(String[] args)
95     {
96         //for(int i=1, int j=10;i<j;i++,j--)
97         {
98             //System.out.print(i+j+" ");
99         }
100    }
101 }
102 /*
103 OUTPUT
104 Compiler error
105 */
106 /*
107 -----
108 /* #6 In the initialization section we can take any valid java statement including
109 sopln.*/
110 //import java.util.*;
111 class Test6
112 {
113     public static void main(String[] args)
114     {
115         int i=0;
116         for(System.out.print("HELLO");i<3;i++)
117         {
118             System.out.print(" HI");
119         }
120     }
121 }
122 /*
123 OUTPUT
124 HELLO HI HI HI
125 */
126 /*
127 -----
128 /* #7 In the Conditional section we can take any valid java statement but it should be
129 of boolean type. If we will not take anything in the conditional section then compiler
130 will place true in the conditional statement.*/
131 //import java.util.*;
132 class Test7
133 {
134     public static void main(String[] args)
135     {
136         for(int i=1;;i++)
137     }

```

```

138         {
139             System.out.print("HELLO");
140         }
141     }
142 }
143 */
144 /*OUTPUT
145 HELLO will be printed infinite times.
146 */
147 /*-----
148
149 /* #8 In the Increment/Decrement section we can take any valid java statement including
150 sopln. We can take multiple increment/decrement variables also.*/
151 //import java.util.*;
152 class Test8
153 {
154     public static void main(String[] args)
155     {
156         int i=0;
157         for(System.out.print("HELLO");i<3;System.out.print(" HI"))
158         {
159             i++;
160         }
161     }
162 }
163 */
164 /*OUTPUT
165 HELLO HI HI HI.
166 */
167 /*-----
168
169 /* #9 All the three parts of for loop are independent of each other and are optional.*/
170 //import java.util.*;
171 class Test9
172 {
173     public static void main(String[] args)
174     {
175         for(;;)
176         {
177         }
178     }
179 }
180 */
181 /*OUTPUT
182 It is infinite loop and will not display anything.
183 */
184 /*-----
185
186
187 /* #10 All the three parts of for loop are independent of each other and are optional.*/
188 //import java.util.*;
189 class Test10
190 {
191     public static void main(String[] args)
192     {
193         for(;;);
194     }
195 }
196 */
197 /*OUTPUT
198 It is infinite loop and will not display anything.
199 */
200 /*-----
201
202 /* #11 Unreachability: in the following java parogram the body of the for loop will be
203 executed infinite times as in the conditional section true is written so the statements
204 after the body of the for loop will never get the chance for its execution. This concept
205 is called Unreachability. And in this case compiler will give compiler Error.*/
206 //import java.util.*;

```

```

207 class Test11
208 {
209     public static void main(String[] args)
210     {
211         for(int i=0;true;i++)
212         {
213             System.out.print("HELLO");
214         }
215         // System.out.print("HI");
216     }
217 }
218 /*OUTPUT
219 error: unreachable statement
220         System.out.print("HI");
221         ^
222 */
223 /*-----*/
224
225 /* #12 Unreachability: in the following java parogram the body of the for loop will not
226 be
227 executed as in the conditional section false is written so the statements within the
228 body
229 of the for loop will never get the chance for its execution. This concept is called
230 Unreachability. And in this case compiler will give compiler Error.*/
231 //import java.util.*;
232 class Test12
233 {
234     public static void main(String[] args)
235     {
236         /* Uncomment this block and see the result
237         for(int i=0;false;i++)
238         {
239             System.out.print("HELLO");
240         }
241         */
242         System.out.print("HI");
243     }
244 }
245 /*OUTPUT
246 error: unreachable statement
247         ^
248 */
249 */
250 /*-----*/
251
252 /* #13 In the above two java parogram in conditional section boolean values true/false are
253 written which are known to compiler that's why it will give the concept of unreachability
254 but in case of any other valid condition in conditional section compiler will not
255 evaluate
256 its boolean value at compile time hence in this case it will not give the concept of
257 unreachability.*/
258 //import java.util.*;
259 class Test13
260 {
261     public static void main(String[] args)
262     {
263         int a=10,b=20;
264         for(int i=0;a<b;i++)
265         {
266             System.out.print("HELLO");
267         }
268         System.out.print("HI");
269     }
270 }
271 /*OUTPUT

```

```

273 Hello will be printed infinite times.
274 */
275 /*-----
276
277 /* #14 In the above two java parogram (Concept 11 and 12) in conditional section boolean
278 values true/false are written which are known to compiler that's why it will give the
279 concept
280 of unreachability but in case of any other valid condition in conditional section
281 compiler
282 will not evaluate its boolean value at compile time hence in this case it will not give
283 the
284 concept of unreachability.*/
285 //import java.util.*;
286 class Test14
287 {
288     public static void main(String[] args)
289     {
290         int a=10,b=20;
291         for(int i=0;a>b;i++)
292         {
293             System.out.print("HELLO");
294         }
295         System.out.print("HI");
296     }
297 /*OUTPUT
298 HI.
299 */
300 /*while loop: while loop is the best choice if we do not know the number of iterations
301 in advance.
302 It is also called entry controlled loop.
303 Syntax:
304 initialization
305 while(Condition of boolean type)
306 {
307     Statement or code to be executed;
308     Increment/Decrement;
309 }
310 */
311 /*-----*/
312 /*#15 WAP to print numbers from 1 to 10 using while loop.*/
313 //import java.util.*;
314 class Test15
315 {
316     public static void main(String[] args)
317     {
318         int i=1;
319         while(i<=10)
320         {
321             System.out.print(i+" ");
322             i++;
323         }
324     }
325 }
326 /* OUTPUT
327 1 2 3 4 5 6 7 8 9 10
328 */
329 /*-----*/
330
331 /* #16 Curly Braces are optional. we can take only one statement under while loop
332 without curly braces. */
333 //import java.util.*;
334 class Test16
335 {
336     public static void main(String[] args)
337     {

```

```

338     while(true)
339         System.out.print("Hello ");
340
341     }
342 }
343 /* OUTPUT
344 Hello will be printed infinite times.
345 */
346
347 /*-----*/
348
349 /* #17 Curly Braces are optional. we can take only one statement under while loop
350 without curly braces which should not be declarative statement. */
351 //import java.util.*;
352 class Test17
353 {
354     public static void main(String[] args)
355     {
356         //while(true)
357         // int x;
358
359     }
360 }
361 /* OUTPUT
362 L1.java:357: error: variable declaration not allowed here
363             int x;
364                     ^
365 1 error
366 */
367 /*-----*/
368
369 /* #18 We can take declarative statement under while loop using curly Braces. */
370 //import java.util.*;
371 class Test18
372 {
373     public static void main(String[] args)
374     {
375         while(true)
376         {
377             int x=10;
378         }
379
380     }
381 }
382 /* OUTPUT
383 No compiler error
384 */
385 /*-----*/
386
387 /* #19 Unreachability: in the following java parogram the body of the while loop will be
388 executed infinite times as in the conditional section true is written so the statements
389 after the body of the while loop will never get the chance for its execution. This
390 concept
391 is called Unreachability. And in this case compiler will give compiler Error.*/
392 //import java.util.*;
393 class Test19
394 {
395     public static void main(String[] args)
396     {
397         while(true)
398         {
399             System.out.print("HELLO");
400         }
401         //System.out.print("HI");
402     }
403 }
404 /*OUTPUT
405 L1.java:400: error: unreachable statement

```

```

406             System.out.print("HI");
407             ^
408         */
409         *-----*/
410
411     /* #20 Unreachability: in the following java parogram the body of the while loop will
412     not be
413     executed as in the conditional section false is written so the statements within the
414     body
415     of the while loop will never get the chance for its execution. This concept is called
416     Unreachability. And in this case compiler will give compiler Error.*/
417     //import java.util.*;
418     class Test20
419     {
420         public static void main(String[] args)
421         {
422             /* Uncomment this block and see the result
423             while(false)
424             {
425                 System.out.print("HELLO");
426             }
427             */
428             System.out.print("HI");
429         }
430     /*OUTPUT
431     L1.java:422: error: unreachable statement
432         {
433             ^
434         */
435     *-----*/
436
437     /* #21 In the above two java parogram in conditional section boolean values true/false are
438     written which are known to compiler that's why it will give the concept of unreachability
439     but in case of any other valid condition in conditional section compiler will not
440     evaluate
441     its boolean value at compile time hence in this case it will not give the concept of
442     unreachability.*/
443     //import java.util.*;
444     class Test21
445     {
446         public static void main(String[] args)
447         {
448             int a=10,b=20;
449             while(a<b)
450             {
451                 System.out.print("HELLO");
452             }
453
454             System.out.print("HI");
455         }
456     }
457     /*OUTPUT
458     Hello will be printed infinite times.
459     */
460     *-----*/
461
462     /* #22 In the above two java parogram (Concept 19 and 20) in conditional section boolean
463     values true/false are written which are known to compiler that's why it will give the
464     concept
465     of unreachability but in case of any other valid condition in conditional section
466     compiler
467     will not evaluate its boolean value at compile time hence in this case it will not give
468     the
469     concept of unreachability.*/
470     //import java.util.*;
471     class Test22

```

```

469 {
470     public static void main(String[] args)
471     {
472         int a=10,b=20;
473         while(a>b)
474         {
475             System.out.print("HELLO");
476         }
477
478         System.out.print("HI");
479     }
480 }
481 /*OUTPUT
482 HI
483 */
484 /*do while loop: If you want to execute loop body atleast once then you should go for
485 do while loop.
486 It is also called exit controlled loop.
487 Syntax:
488 Initialization
489 do
490 {
491     Statement or code to be executed;
492     Increment/Decrement;
493 }while(boolean type condition);
494 */
495 /*
496 -----*/
497
498 /*#23 WAP to print numbers from 1 to 10 using do while loop.*/
499 //import java.util.*;
500 class Test23
501 {
502     public static void main(String[] args)
503     {
504         int i=1;
505         do
506         {
507             System.out.print(i+" ");
508             i++;
509         }while(i<=10);
510     }
511 }
512 /* OUTPUT
513 1 2 3 4 5 6 7 8 9 10
514 */
515 /*
516 -----*/
517 /* #24 Curly Braces are optional. we can take only one statement under do while loop
518 without curly braces. */
519 //import java.util.*;
520 class Test24
521 {
522     public static void main(String[] args)
523     {
524         do
525             System.out.print("Hello ");
526
527         while(true);
528     }
529 }
530 /* OUTPUT
531 Hello will be printed infinite times.
532 */
533 /*
534 -----*/
535 /* #25 Curly Braces are optional. we can take only one statement under do while loop
536 without curly braces which should not be declarative statement. */
537 //import java.util.*;
```

```

538 class Test25
539 {
540     public static void main(String[] args)
541     {
542         //do
543         // int x;
544         //while(true);
545     }
546 }
547 /* OUTPUT
548 L1.java:543: error: variable declaration not allowed here
549             int x;
550                     ^
551 1 error
552 */
554 /*-----*/
555
556 /* #26 We can take declarative statement under do while loop using curly Braces. */
557 //import java.util.*;
558 class Test26
559 {
560     public static void main(String[] args)
561     {
562         do
563         {
564             int x=10;
565         }
566         while(true);
567     }
568 }
569 /* OUTPUT
570 No compiler error
571 */
573 /*-----*/
574
575 /* #27 Unreachability: in the following java parogram the body of the do while loop
576 will be
577 executed infinite times as in the conditional section true is written so the statements
578 after the body of the do while loop will never get the chance for its execution. This
579 concept
580 is called Unreachability. And in this case compiler will give compiler Error.*/
581 //import java.util.*;
582 class Test27
583 {
584     public static void main(String[] args)
585     {
586         do
587         {
588             System.out.print("HELLO");
589             }while(true);
590             //System.out.print("HI");
591     }
592 /*OUTPUT
593 L1.java:588: error: unreachable statement
594             System.out.print("HI");
595                     ^
596 1 error
597 */
598 /*-----*/
599
600 /* #28 What will be the output of following java program?*/
601 //import java.util.*;
602 class Test28
603 {
604     public static void main(String[] args)

```

```

605     {
606
607     do
608     {
609         System.out.print("HELLO ");
610         }while(false);
611
612         System.out.print("HI");
613
614     }
615 }
616 /*OUTPUT
617 HELLO HI
618 */
619 /*-----
620
621 /* #29 In the above 27th java parogram in conditional section boolean value true is
622 written which is known to compiler that's why it will give the concept of unreachability
623 but in case of any other valid condition in conditional section compiler will not
624 evaluate
625 its boolean value at compile time hence in this case it will not give the concept of
626 unreachability.*/
627 //import java.util.*;
628 class Test29
629 {
630     public static void main(String[] args)
631     {
632         int a=10,b=20;
633         do
634         {
635             System.out.print("HELLO ");
636             }while(a<b);
637
638             System.out.print("HI");
639     }
640 }
641 /*OUTPUT
642 Hello will be printed infinite times.
643 */
644 /*-----
645
646 /* #30 In the above java parogram (Concept 27) in conditional section boolean
647 value true is written which is known to compiler that's why it will give the concept
648 of unreachability but in case of any other valid condition in conditional section
649 compiler
650 will not evaluate its boolean value at compile time hence in this case it will not give
651 the
652 concept of unreachability.*/
653 //import java.util.*;
654 class Test30
655 {
656     public static void main(String[] args)
657     {
658         int a=10,b=20;
659         do
660         {
661             System.out.print("HELLO ");
662             }while(a>b);
663
664             System.out.print("HI");
665     }
666 }
667 /*OUTPUT
668   HELLO HI
669 */
670 /*-----*/

```

```
671 /*
672 Transfer Statements:
673 1. break: #31 The break statement is used to terminate the loop immediately.
674 We can use break statement inside switch and inside the loops.
675 */
676 class Test31
677 {
678     public static void main(String[] args)
679     {
680         for(int i=1;i<=10;i++)
681         {
682             if(i==5)
683             {
684                 break;
685             }
686             else
687             {
688                 System.out.print(i+ " ");
689             }
690         }
691     }
692 }
693 /*OUTPUT
694 1 2 3 4
695 */
696 /*-----*/
697 /*
698 Transfer Statements:
699 2. continue: #32 The continue statement is used to skip the current iteration of the
700 loop.
701 We can use continue statement inside the loops.
702 */
703 class Test32
704 {
705     public static void main(String[] args)
706     {
707         for(int i=1;i<=10;i++)
708         {
709             if(i==5)
710             {
711                 continue;
712             }
713             else
714             {
715                 System.out.print(i+ " ");
716             }
717         }
718     }
719 }
720 /*OUTPUT
721 1 2 3 4 6 7 8 9 10
722 */
```