

Nombre de la práctica	Ejercicios en Lenguaje C			No.	3
Asignatura:	Métodos Numéricos	Carrera:	ISIC	Duración de la práctica (Hrs)	

Nombre: Rodrigo Javier Martínez

Grupo: 3041

## I. Competencia(s) específica(s):

## II. Lugar de realización de la práctica (laboratorio, taller, aula u otro):

- Aula y casa

## III. Material empleado:

- Dev-C++
- Word

## IV. Desarrollo de la práctica:

Esta es la continuación del curso de Lenguaje C que llevamos a cabo en Métodos Numéricos, en donde estamos realizando ejercicios en este lenguaje.

Para continuar con estos ejercicios vamos a ver que es un ciclo while, el propósito de este ciclo es repetir un bloque de código mientras una condición se mantenga verdadera.

¿Cómo funciona?

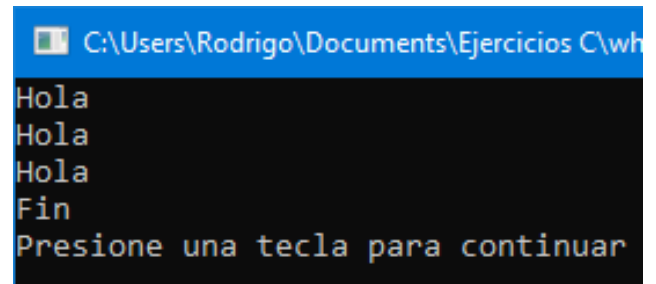
Verifica si la condición se cumple si es verdadero, ejecuta una o varias instrucciones y nuevamente verifica la condición

Si es falsa, entonces el ciclo termina.

En el siguiente programa solo se mandó a imprimir un mensaje el cual se va a imprimir las veces que la condición del ciclo indica.

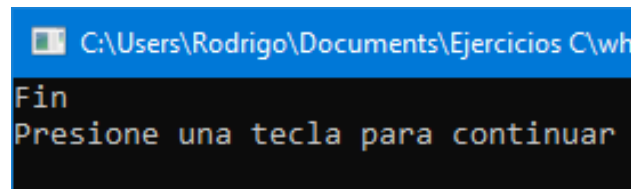
```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main(){
5      int contador = 0;
6      while(contador<3){
7          printf("Hola\n");
8          contador++;
9      }
10     printf("Fin\n");
11     system("pause");
12 }
```



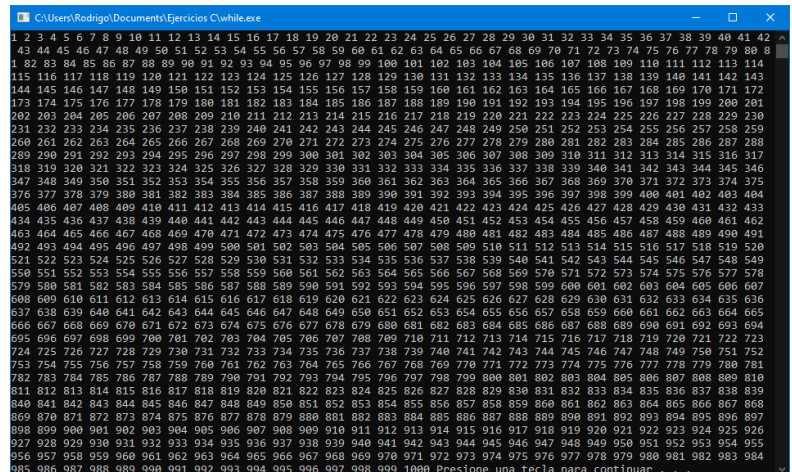
Este programa es similar al anterior solo que se modificó el contador, en donde este es mayor que la condición del ciclo así que no podrá acceder a este ciclo.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(){
5     int contador = 5;
6     while(contador<3){
7         printf("Hola\n");
8         contador++;
9     }
10    printf("Fin\n");
11    system("pause");
12 }
```



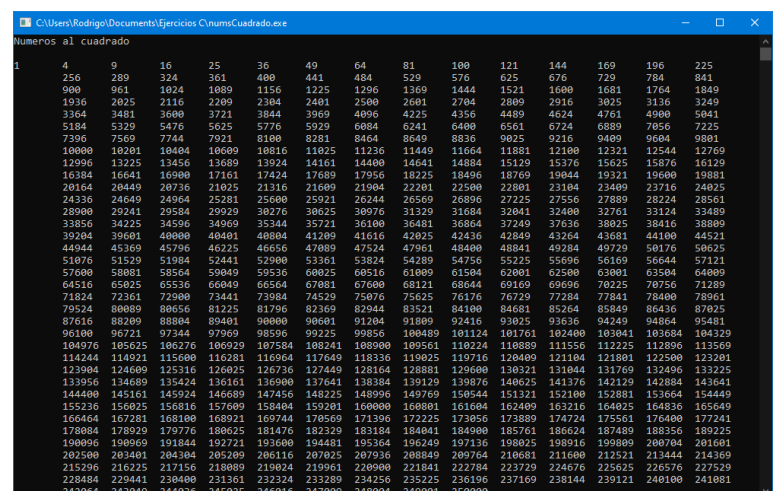
Para este programa se va imprimir los números del 1 al 1000 con un ciclo while.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(){
5     int numero = 1;
6
7     while(numero<=1000){
8         printf("%d ", numero);
9         numero++;
10    }
11    system("pause");
12    return 0;
13 }
```



Para el siguiente programa se debe de imprimir los cuadrados y los cubos de los primeros quinientos números naturales con un ciclo while.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(){
5
6     int numeroCuadrado = 1;
7     int numeroCubo = 1;
8     puts("Numeros al cuadrado\n");
9     while(numeroCuadrado<=500){
10        printf("%d\t", numeroCuadrado*numeroCuadrado);
11        numeroCuadrado++;
12    }
13    puts("\nNumeros al cubo\n");
14    while(numeroCubo<=500){
15        printf("%d\t", numeroCubo*numeroCubo*numeroCubo);
16        numeroCubo++;
17    }
18    system("pause");
19    return 0;
20 }
```



C:\Users\Rodrigo\Documents\Ejercicios\C\numsCuadrado.exe

Numeros al cubo

1	8	27	64	125	216	343	512	729	1000	1331	1728	2197	2744	3375
	4096	4913	5832	6859	8000	9261	10648	12167	13824	15625	17576	19683	21952	24389
	27000	29791	32768	35937	39304	42875	46656	50653	54872	59319	64000	68921	74088	79507
	85184	91125	97336	103823	110592	117649	125000	132651	140608	148877	157464	166375	175616	185193
	195112	205379	216000	226981	238328	250047	262144	274625	287496	300763	314432	328509	343000	357911
	373248	389017	405224	421875	438976	456533	474552	493039	512000	531441	551368	571787	592704	614125
	636056	658503	681472	704969	729000	753571	778688	804357	830584	857375	884736	912673	941192	970299
	1000000	1030301	1061208	1092727	1124864	1157625	1191016	1225043	1259712	1295029	1331000	1367631	1404928	1442897
	1481544	1520875	1560896	1601613	1643032	1685159	1728000	1771561	1815848	1860867	1906624	1953125	2000376	2048383
	2097152	2146689	2197000	2248091	2299968	2352637	2406104	2460375	2515456	2571353	2628072	2685619	2744000	2803221
	2863288	2924207	2985984	3048625	3112136	3176523	3241792	3307940	3375000	3442951	3511808	3581577	3652264	3723875
	3796416	3868083	3940312	4013117	4086504	4160479	4235040	4310193	4385944	4462297	4539256	4616825	4694999	4773784
	4913000	5000211	5088448	5177717	5268024	5359375	5451776	5545233	5639752	5735339	5832000	5929741	6028568	6128487
	6229504	6331625	6434856	6539203	6644672	6751269	6859000	6967871	7077888	7189057	7301384	7414875	7529536	7645373
	7762392	7880599	8000000	8120601	8242408	8365427	8489664	8615125	8741816	8869743	8998912	9129329	9261000	9393931
	9528128	9663597	9800344	9938375	10077696	10218313	10360232	10503459	10648000	10793871	10941088	11089647	11239552	11390809
	10793861	10941088	11089647	11239552	11390809	11543176	11696883	11851952	12008400	12166224	12325441	12486048	12648048	12811449
	12977875	13144256	13312053	13481272	13651919	13824000	13997521	14174576	14353125	14533176	14714736	14897809	15082392	15268492
	15454976	15642609	15831751	16022400	16214561	16408224	16603399	16800080	16998267	17197960	17399161	17601872	17806092	18011824
	18219072	18427329	18636592	18846861	19058128	19270399	19483672	19697949	19913224	20129500	20346776	20565049	20784320	21004592
	21225856	21447125	21669392	21892656	22116919	22342184	22568449	22795712	23023976	23253240	23483504	23714769	23947032	24180292
	24413552	24648809	24885072	25122344	25360616	25600880	25843136	26086384	26330624	26576856	26824080	27072296	27321504	27571704
	27822896	28075089	28328272	28582444	28838599	29095744	29353880	29613000	29873104	30134192	30396264	30659320	30923360	31188384
	31454392	31720432	31987504	32255600	32524720	32794864	33066032	33338224	33611440	33885680	34160944	34437224	34714520	34991832
	35270160	35547760	35826416	36106128	36386896	36668720	36951600	37235536	37520528	37806576	38093680	38381840	38671056	38961328
	39252656	39545888	39840176	40135520	40431920	40729376	41027888	41327456	41628080	41929760	42232496	42536288	42841136	43147040
	43454000	43762720	44072448	44383184	44694928	45007680	45321440	45636208	45951984	46268768	46586560	46905360	47225168	47545984
	47867808	48189568	48512336	48836112	49160896	49486688	49813488	50141296	50470112	50800928	51132744	51465560	51799376	52134192
	52470000	52805808	53142608	53480400	53819184	54158960	54500720	54843472	55187216	55531952	55878672	56226376	56575056	56924704
	57275328	57625056	57975760	58327440	58680096	59033728	59388336	59743920	60100480	60458016	60816528	61176016	61536480	61897920
	62260336	62622784	62986272	63350792	63716336	64082904	64450496	64819112	65188752	65559408	65931080	66303768	66677472	67052192
	67427920	67805328	68183712	68563072	68943408	69324720	69707008	70090272	70474512	70859720	71245896	71633040	72021152	72410224
	72800256	73190288	73581296	73973280	74366240	74760176	75155088	75550976	75947840	76345680	76744496	77144288	77545056	77946792
	78349504	78755776	79163008	79571192	79980328	80390416	80801456	81213448	81626392	82040288	82455136	82870936	83287680	83705376
	84124016	84542768	84962480	85383152	85804784	86227376	86650928	87075440	87500912	87927344	88354736	88783088	89212392	89642656
	89992864	90423968	90856032	91289056	91723040	92157984	92593888	93030752	93468576	93907360	94347104	94787808	95229472	95672096
	96115680	96554304	96993872	97434384	97875840	98318240	98761584	99205888	99651136	100097440	100545696	100994912	101445088	101896224

Con un ciclo while realizar lo siguiente:

Imprimir todos los números divisibles entre 3 mayores a 0 y menores a mil

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(){
5     int num = 1 ;
6     int num1 = 1;
7     int num2 = 1 ;
8     puts("Numeros divisibles de 3\n");
9     while(num<1000){
10         if(num %3 == 0){
11             printf("%d\t",num);
12             num++;
13         }
14     }
```

C:\Users\Rodrigo\Documents\Ejercicios\C\numsDivisibles.exe

Numeros divisibles de 3

6	9	12	15	18	21	24	27	30	33	36	39	42	45
48	51	54	57	60	63	66	69	72	75	78	81	84	87
90	93	96	99	102	105	108	111	114	117	120	123	126	129
132	135	138	141	144	147	150	153	156	159	162	165	168	171
174	177	180	183	186	189	192	195	198	201	204	207	210	213
216	219	222	225	228	231	234	237	240	243	246	249	252	255
258	261	264	267	270	273	276	279	282	285	288	291	294	297
300	303	306	309	312	315	318	321	324	327	330	333	336	339
342	345	348	351	354	357	360	363	366	369	372	375	378	381
384	387	390	393	396	399	402	405	408	411	414	417	420	423
426	429	432	435	438	441	444	447	450	453	456	459	462	465
468	471	474	477	480	483	486	489	492	495	498	501	504	507
510	513	516	519	522	525	528	531	534	537	540	543	546	549
552	555	558	561	564	567	570	573	576	579	582	585	588	591
594	597	600	603	606	609	612	615	618	621	624	627	630	633
636	639	642	645	648	651	654	657	660	663	666	669	672	675
678	681	684	687	690	693	696	699	702	705	708	711	714	717
720	723	726	729	732	735	738	741	744	747	750	753	756	759
762	765	768	771	774	777	780	783	786	789	792	795	798	801
804	807	810	813	816	819	822	825	828	831	834	837	840	843
846	849	852	855	858	861	864	867	870	873	876	879	882	885
888	891	894	897	900	903	906	909	912	915	918	921	924	927
930	933	936	939	942	945	948	951	954	957	960	963	966	969
972	975	978	981	984	987	990	993	996	999				

Escribir todos los enteros positivos menores que 100 omitiendo aquellos que son divisibles por 7.

```
puts("\nNumeros que no sean divisibles de 7\n");
while(num2<100){
    if(num2%7 != 0){
        printf("%d\t",num2);
    }
    num2++;
}
system("pause");
return 0;
```

Numeros que no sean divisibles de 7

1	2	3	4	5	6	8	9	10	11	12	13	15	16	17
18	19	20	22	23	24	25	26	27	29	30	31	32	33	34
35	36	37	38	39	40	41	43	44	45	46	47	48	50	51
52	53	54	55	56	57	58	59	60	61	62	64	65	66	67
68	69	71	72	73	74	75	76	78	79	80	81	82	83	84
85	86	87	88	89	90	92	93	94	95	96	97	99		

Presione una tecla para continuar . . .



Imprimir todos los números que son divisibles entre 2 y entre 7, mayores a 0 y menores a mil.

```
puts("\nNumeros disvibles de 2\n");
while(num1<1000){
    if(num1%2 == 0){
        printf("%d\t",num1);
    }
    num1++;
}
```

```

Numeros disvibles de 2
2
4 6 8 10 12 14 16 18 20 22 24 26 28 30
32 34 36 38 40 42 44 46 48 50 52 54 56 58
60 62 64 66 68 70 72 74 76 78 80 82 84 86
88 90 92 94 96 98 100 102 104 106 108 110 112 114
116 118 120 122 124 126 128 130 132 134 136 138 140 142
144 146 148 150 152 154 156 158 160 162 164 166 168 170
172 174 176 178 180 182 184 186 188 190 192 194 196 198
200 202 204 206 208 210 212 214 216 218 220 222 224 226
228 230 232 234 236 238 240 242 244 246 248 250 252 254
256 258 260 262 264 266 268 270 272 274 276 278 280 282
284 286 288 290 292 294 296 298 300 302 304 306 308 310
312 314 316 318 320 322 324 326 328 330 332 334 336 338
340 342 344 346 348 350 352 354 356 358 360 362 364 366
368 370 372 374 376 378 380 382 384 386 388 390 392 394
396 398 400 402 404 406 408 410 412 414 416 418 420 422
424 426 428 430 432 434 436 438 440 442 444 446 448 450
452 454 456 458 460 462 464 466 468 470 472 474 476 478
480 482 484 486 488 490 492 494 496 498 500 502 504 506
508 510 512 514 516 518 520 522 524 526 528 530 532 534
536 538 540 542 544 546 548 550 552 554 556 558 560 562
564 566 568 570 572 574 576 578 580 582 584 586 588 590
592 594 596 598 600 602 604 606 608 610 612 614 616 618
620 622 624 626 628 630 632 634 636 638 640 642 644 646
648 650 652 654 656 658 660 662 664 666 668 670 672 674
676 678 680 682 684 686 688 690 692 694 696 698 700 702
704 706 708 710 712 714 716 718 720 722 724 726 728 730
732 734 736 738 740 742 744 746 748 750 752 754 756 758
760 762 764 766 768 770 772 774 776 778 780 782 784 786
788 790 792 794 796 798 800 802 804 806 808 810 812 814
816 818 820 822 824 826 828 830 832 834 836 838 840 842
844 846 848 850 852 854 856 858 860 862 864 866 868 870
872 874 876 878 880 882 884 886 888 890 892 894 896 898
900 902 904 906 908 910 912 914 916 918 920 922 924 926
928 930 932 934 936 938 940 942 944 946 948 950 952 954
956 958 960 962 964 966 968 970 972 974 976 978 980 982
984 986 988 990 992 994 996 998

```

El ciclo do-while es muy parecido al while pero esta estructura primero ejecuta el conjunto de instrucciones y después verifica que la condición se cumpla.

¿Cómo funciona?

Realiza es bloque de código que se encuentra en DO

Después verifica si la condición se cumple

Si es verdadera, repite el ciclo

Si es falsa, entonces el ciclo termina.

Un ejemplo es el siguiente en donde se van a imprimir los valores de i, esto se realizará hasta que la condición ya no se cumpla.

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main(){
5      int i = 0;
6      do{
7          printf("Valor de i = %d \n", i);
8          i++;
9      }while(i<3);
10     printf("Fin\n");
11     system("pause");
12
13 }
```

```

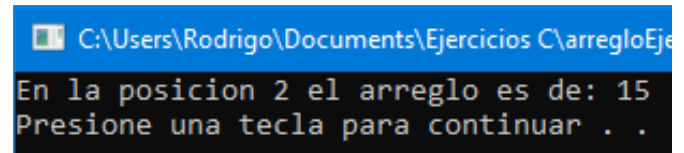
C:\Users\Rodrigo\Documents\Ejercicios C\doWhile
Valor de i = 0
Valor de i = 1
Valor de i = 2
Fin
Presione una tecla para continuar .

```

Los arreglos son variables que hace referencia a varias posiciones de memoria. Cada posición se identifica con un índice, el índice comienza en 0.

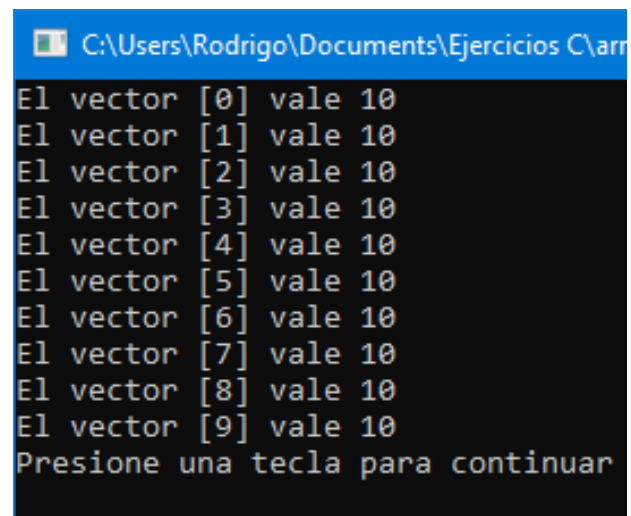
El siguiente programa es un ejemplo de cómo declarar un arreglo y asignarle un valor en alguna posición y sumar estas dos posiciones.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(){
5     int miArreglo[8];
6
7     miArreglo[0]=5;
8     miArreglo[1]=10;
9     miArreglo[2]= miArreglo[0]+miArreglo[1];
10    printf("En la posicion 2 el arreglo es de: %d \n", miArreglo[2]);
11    system("pause");
12    return 0;
13 }
```



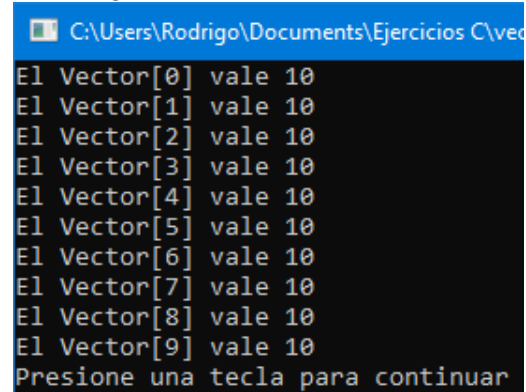
Crear un programa que declare un arreglo llamado "vector" de 10 posiciones. Asignar el valor de 10 a cada posición del arreglo e imprimir todas las posiciones del arreglo.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(){
5     int vector[10];
6     vector[0]= 10;
7     vector[1]= 10;
8     vector[2]= 10;
9     vector[3]= 10;
10    vector[4]= 10;
11    vector[5]= 10;
12    vector[6]= 10;
13    vector[7]= 10;
14    vector[8]= 10;
15    vector[9]= 10;
16
17    printf("El vector [0] vale %d\n", vector[0]);
18    printf("El vector [1] vale %d\n", vector[1]);
19    printf("El vector [2] vale %d\n", vector[2]);
20    printf("El vector [3] vale %d\n", vector[3]);
21    printf("El vector [4] vale %d\n", vector[4]);
22    printf("El vector [5] vale %d\n", vector[5]);
23    printf("El vector [6] vale %d\n", vector[6]);
24    printf("El vector [7] vale %d\n", vector[7]);
25    printf("El vector [8] vale %d\n", vector[8]);
26    printf("El vector [9] vale %d\n", vector[9]);
27    system("pause");
28    return 0;
29 }
```



El programa anterior se puede hacer con while y quedaria de la siguiente manera:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(){
5     int vector[10];
6     int i=0;
7     while(i<10){
8         vector[i]=10;
9         i++;
10    }
11    i=0;
12    while(i<10){
13        printf("El Vector[%d] vale %d\n",i,vector[i]);
14        i++;
15    }
16    system("pause");
17    return 0;
18 }
```





Crear un arreglo de 100 posiciones, llenar el arreglo con la tabla del 2 e imprimir el arreglo en pantalla.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(){
5     int vector[100];
6     int oper;
7     int i=1;
8
9     while(i<100){
10         vector[i]=2;
11         oper = i*2;
12         i++;
13         printf("%d\t",oper);
14     }
15     system("pause");
16     return 0;
17 }
```

C:\Users\Rodrigo\Documents\Ejercicios C\arregloTabla2.exe

2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
	32	34	36	38	40	42	44	46	48	50	52	54	56	58
	60	62	64	66	68	70	72	74	76	78	80	82	84	86
	88	90	92	94	96	98	100	102	104	106	108	110	112	114
	116	118	120	122	124	126	128	130	132	134	136	138	140	142
	144	146	148	150	152	154	156	158	160	162	164	166	168	170
	172	174	176	178	180	182	184	186	188	190	192	194	196	198

Crear un arreglo de 100 posiciones, que se llene en orden inverso al índice e imprima el arreglo en pantalla.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(){
5     int vector[100];
6     int i;
7     for(i=1;i<=100;i++){
8         vector[i]=i;
9     }
10    for(i=100;i>=1;i--){
11        printf("%d\t", i,vector[i]);
12    }
13    system("pause");
14    return 0;
15 }
```

C:\Users\Rodrigo\Pictures\Camera Roll\arregloInverso.exe

100	99	98	97	96	95	94	93	92	91	90	89	88	87	86
	85	84	83	82	81	80	79	78	77	76	75	74	73	72
	71	70	69	68	67	66	65	64	63	62	61	60	59	58
	57	56	55	54	53	52	51	50	49	48	47	46	45	44
	43	42	41	40	39	38	37	36	35	34	33	32	31	30
	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2
1	Presione una tecla para continuar . . .													

En el siguiente programa se realizarán operaciones con dos arreglos establecidos:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(){
5
6     int A[]={3,5,6,8,4,7,8,5,3,1};
7     int B[]={3,4,6,8,9,1,2,3,0,9};
8
9     printf("%d\n", A[3]*(B[2]/2));
10    printf("%d\n",B[A[1]]-A[9]);
11    printf("%d\n",A[0]+A[1+2]);
12    printf("%d\n", A[5]+B[5]);
13    printf("%d\n", (A[3]/B[2]/2));
14
15    system("pause");
16    return 0;
17 }
```

C:\Users\Rodrigo\Documents\Ejercicios C\arre

```
2
0
11
8
0
Presione una tecla para continuar
```

Crea un arreglo de 20 posiciones, asígnale a cada elemento un valor, calcula el promedio de todos los elementos y calcula la multiplicación de todos los elementos.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(){
5
6     int a[]={3,5,6,8,4,7,8,5,3,1,9,7,2,8,15,5,1,4,10,2};
7     double prom;
8     long mult;
9     prom = (a[0]+a[1]+a[2]+a[3]+a[4]+a[5]+a[6]+a[7]+a[8]
10    +a[9]+a[10]+a[11]+a[12]+a[13]+a[14]+a[15]+a[16]+a[17]+a[18]+a[19])/20;
11    printf("El promedio es: %f\n", prom);
12    mult = (a[0]*a[1]*a[2]*a[3]*a[4]*a[5]*a[6]*a[7]*a[8]
13    *a[9]*a[10]*a[11]*a[12]*a[13]*a[14]*a[15]*a[16]*a[17]*a[18]*a[19]);
14    printf("El resultado de la multiplicacion es: %d\n", mult);
15    system("pause");
16    return 0;
17 }
```

```
C:\Users\Rodrigo\Documents\Ejercicios C\arreglo20.exe
El promedio es: 5.000000
El resultado de la multiplicacion es: 1631977472
Presione una tecla para continuar . . .
```

Un ciclo For inicializa el valor del contador, verifica si la condición se cumple y en tal caso ejecuta las instrucciones, posteriormente incrementa o decrementa la variable contador.

Este es un ejemplo de como implementar un ciclo for, en este program lo realizamos con una serie del numero 2 y quedo de la siguiente manera.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(){
5     int longitudSerie=50;
6     int i;
7
8     for(i = 1;i<(longitudSerie/2);i++){
9         printf("%d\t", 2*i);
10        printf("%d\t", 3*i);
11    }
12    system("pause");
13    return 0;
14 }
```

```
C:\Users\Rodrigo\Documents\Ejercicios C\ejemploFor.exe
2 3 4 6 6 9 8 12 10 15 12 18 14 21 16
24 18 27 20 30 22 33 24 36 26 39 28 42 30
45 32 48 34 51 36 54 38 57 40 60 42 63 44
66 46 69 48 72 Presione una tecla para continuar . . .
```

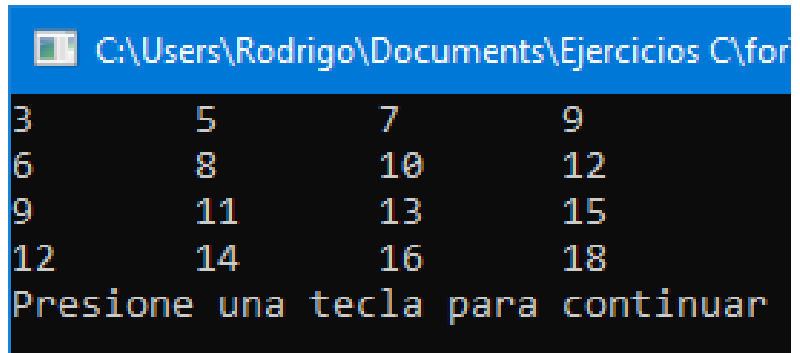
Escribe un programa que reciba un número N del usuario y haga la suma de todos los números desde 1 hasta N

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(){
5
6     int nums;
7     puts("Ingresa la cantidad de numeros que desea");
8     scanf("%d", &nums);
9     int i=1,j=0;
10    for(i = 0;i<=nums;i++){
11        j=j+i;
12    }
13    printf("El resultado es: %d\n",j);
14    system("pause");
15    return 0;
16 }
```

```
C:\Users\Rodrigo\Documents\Ejercicios C\cicloFor.exe
Ingresa la cantidad de numeros que desea
5
El resultado es: 15
Presione una tecla para continuar . . .
```

Escriba un programa en C que utilice un ciclo para producir una tabla la cual valla incrementando su valor.

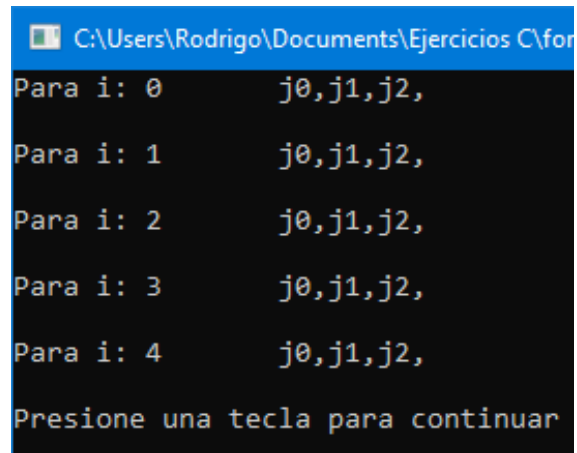
```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main(){
5      int i;
6      for(i=2;i<9;i++){
7          i=i+1;
8          printf("%d\t",i);
9      }
10     printf("\n");
11     int j;
12     for(j=5;j<12;j++){
13         j=j+1;
14         printf("%d\t",j);
15     }
16     printf("\n");
17     int k;
18     for(k=8;k<15;k++){
19         k=k+1;
20         printf("%d\t",k);
21     }
22     printf("\n");
23     int l;
24     for(l=11;l<18;l++){
25         l=l+1;
26         printf("%d\t",l);
27     }
28     printf("\n");
29     system("pause");
30     return 0;
31 }
```



Un for anidado es un ciclo dentro de otro, el primer ciclo esperara a que el segundo ciclo acabe su función y se ejecutara el primer ciclo hasta que la condición establecida ya no se cumpla.

Un ejemplo es el siguiente programa en donde se empleó un ciclo anidado.

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main (){
5      int i;
6      int j;
7
8      for(i=0;i<5;i++){
9
10         printf("Para i: %d \t", i);
11         for(j=0;j<3;j++){
12             printf("j%d,", j);
13         }
14         printf("\n\n");
15     }
16     system("pause");
17     return 0;
18 }
```









## V. Conclusiones:

Para concluir podemos decir que el lenguaje de programación C es uno de los más completos y complejos de utilizar ya que cuenta con muchas funciones las cuales nos pueden ayudar a resolver distintos problemas ya que los puedes implementar en la programación y sacarle provecho a esta. También agregar que el lenguaje C es la base casi toda la programación que su estructura es muy similar a las de los demás lenguajes.