	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor:	Adrian Ulises Mercado Martinez
Asignatura:	Fundamentos de programación.
Grupo:	10
No de Práctica(s):	7
Integrante(s):	González Bretón Arturo Paniagua Bautista Daniel Castañeda Mora Carlos
No. de Equipo de cómputo empleado:	49-50-51
No. de Lista o Brigada:	14
Semestre:	1
Fecha de entrega:	19/09/19
Observaciones:	

CALIFICACIÓN: _____

Práctica7Desarrollo de programas en C

Castañeda Mora Carlos
González Bretón Arturo
Paniagua Bautista Daniel

26 Septiembre 2019

1 Introduction

Los compiladores reservan determinados términos ó palabras claves (keywords), para el uso sintáctico del lenguaje, tales como: asm, auto, break, case, char, do, for, etc. Si bien estas palabras están definidas para el ANSI C, los distintos compiladores extienden esta definición a otros términos, por lo que es aconsejable leer la tabla completa de palabras reservadas del compilador que se vaya a usar, para no utilizarlas en nombres de variables.

Para crear una variable en un lugar determinado del un programa escribiremos primero el tipo de variable y luego el identificador con el que queremos nombrar la variable, seguido todo de un ';'. A esto se le denomina definir una variable. La forma general de la definición es:

tipo lista$_i dentificadores$; $Porejemplo$:

int numero; crea la variable numero, de tipo número entero char letra; crea la variable letra, de tipo carácter float a, b; crea dos variables a y b, de tipo número de coma flotante Las variables se pueden inicializar, es decir, establecer un valor inicial, en el momento de creación. Para ello, detrás del identificador ponemos el carácter '=' seguido del valor inicial. Los valores iniciales pueden ser cualquier constante válida para el tipo de variable que creemos. Por ejemplo:

int numero = 0; crea la variable entera numero y la inicializa a 0 char letra = 'p'; crea la variable carácter letra y la inicializa a 'p' float a=57.1, b=1.2E-5; crea las variables a y b, de tipo número de coma flotante, y los inicializa a 57.1 y 0.000012, respectivamente

2 Desarrollo

1)Valores de tipo entero.

```c
#include <stdio.h>
#include<stdlib.h>

int main(){

        short _; //Variable entera con espacio de almacenaje pequeño.
        signed int __;//Variable entera con signo.
        int v_1;//Variable entera con signo.
        unsigned int v_2;//Variable entera sin signo.
        long v_3//Variable entera con mucho espacio de almacenamiento.

        return 0;

}
~
```

```
Ingresa un dato de tipo short
33000
Ingresa un dato de tipo signed int
2147483647
Ingresa un dato de tipo int
2147483647
Ingresa un dato de tipo unsigned int
2147483646
Ingresa un dato de tipo long
92233393939393
Valores leídos
-32536  2147483647      2147483647      2147483646      92233393939393
Guadalupe49:~ fp15alu11$
```

```c
#include <stdio.h>
#include<stdlib.h>

int main(){

        short _; //Variable entera con espacio de almacenaje pequeño.
        signed int __;//Variable entera con signo.
        int v_1;//Variable entera con signo.
        unsigned int v_2;//Variable entera sin signo.
        long v_3;//Variable entera con mucho espacio de almacenamiento.
        printf("Ingresa un dato de tipo short\n");
        scanf("%hd",&_);
        printf("Ingresa un dato de tipo signed int\n");
        scanf("%d",&__);
        printf("Ingresa un dato de tipo int\n");
        scanf("%d",&v_1);

        printf("Ingresa un dato de tipo unsigned int\n");
        scanf("%d",&v_2);
        printf("Ingresa un dato de tipo long\n");
        scanf("%ld",&v_3);
        printf("Valores leídos\n");
        printf("%hd\t%d\t%d\t%ld\n",_,__,v_1,v_2,v_3);
        return 0;
```

2)Valores de tipo decimal.

```c
#include <stdio.h>
#include<stdlib.h>

int main(){

        float v_f;
        double v_d;
        long double v_ld;
        printf("Ingresa un dato de tipo float\n");
        scanf("%hd",&v_f);
        printf("Ingresa un dato de tipo double \n");
        scanf("%d",&v_d);
        printf("Ingresa un dato de tipo long double\n");
        scanf("%d",&v_ld);

                printf("Valores leídos\n");
        printf("%f\t%e\t%g\n",v_f,v_f,v_f);
        printf("%f\t%e\t%g\n",v_d,v_d,vd);
        printf("%f\t%e\t%g\n",v_ld,v_ld,v_ld);
        return 0;

}
~
```

3)Variable de tipo Bool y caracter.

```c
#include <stdio.h>
#include<stdlib.h>
#include <stdbool.h>

int main(){

        char v1;//variable tipo char
        bool v2;//variable tipo bool
        char v3[10];//cadena de 10 caracteres
        printf("Ingresa un dato de tipo char\n");
        scanf("%c",&v1);
        printf("Ingresa un dato de tipo bool \n");
        scanf("%d",&v2);
        printf("Ingresa un dato de tipo cadena de char\n");
        scanf("%s",&v3);

                printf("Valores leídos\n");
        printf("%c",v1);
        printf("%d",v2);
        printf("%s",v3);
        return 0;

}
```

4)Comparaciòn de nùmeros.

```c
#include <stdio.h>

int main(){
        int a,b,c,d,f;
        short g,h,i,j;

        a = 5;
        b=8;
        c=10;
        d=12;
        f=45;

        printf("a > b= %d\n", a>b);
        printf("a < b= %d\n", a<b);
        printf("a >= b= %d\n", a>=b);
        printf("a <= b= %d\n", a<=b);
        printf("a == b= %d\n", a==b);
        printf("a != b= %d\n", a!=b);


        printf("a && b= %d\n", a&&b);
        printf("a || b= %d\n", a||b);
        printf("!a = %d\n", !a);
        g=16;
        h=8;
        i=4;
        j=2;
        printf("g>>h= %d\n",g>>h);
        printf("g>>j= %d\n",g>>j);
        printf("i<<j= %d\n",i<<j);
        printf("i<<h= %d\n",i<<h);
        return 0;

}
```

```
Guadalupe49:~ fp15alu11$ ./programa1
a > b= 0
a < b= 1
a >= b= 0
a <= b= 1
a == b= 0
a != b= 1
a && b= 1
a || b= 1
!a = 0
g>>h= 0
g>>j= 4
i<<j= 16
i<<h= 1024
Guadalupe49:~ fp15alu11$
```

3 Conclusion

Al escribir el código es necesario identificar con claridad qué tipo de datos son los adecuados para resolver el problema ya que cada una tiene características que limitan su funcionamiento, por ejemplo un dato tipo int tiene un rango de -2,147,483,638 a 2,147,483,637, pero si sobrepasamos este valor obenemos 2,147,483,637, es decir, que si el número que utilizamos está fuera de rango el intérprete nos proporciona otro, esto producirá errores en el resultado final. En lenguaje C existen tipos de datos con nombres diferentes como signed int e int, que realizan la misma función. Daniel

En está práctica aprendimos a diferenciar los tipos de variables que encontramos en un código, para administrar correctamente la memoria y no quedarnos

cortos al manejar variables demasiado grandes, por ejemplo, cuando sobre pasas el máximo de una variable, sólo representa el número mayor, por lo que perderíamos información,que en algunos casos puede ser fatal. Tambien es necesario optimizar este tipo de variables, por ejemplo, si los números con los que vamos a trabajar tienen pocos dígitos, no es necesario declarar un tipo long, pues estás reservando un trozo más grande dde la memoria, en pocas palabras, no estás optimizando tu programa. Carlos.

4 Bibliografía

https://es.wikipedia.org/wiki/C$_(lenguaje_de_programacihttp://platea.pntic.mec.es/vgonzale/cyr_0204/cyr_01/$