**Ejemplo 1**

**Apuntadores**

#include <stdio.h>

int main(){

int var1 = 10;

int \*apEntero = &var1;

printf ("Valor de var1 : %d", var1);

printf("\n\nValor de apEntero: %d", apEntero);

printf("\n\nDireccion de memoria de var1 : %d", &var1);

printf("\n\nValor de \*apEntero : %d", \*apEntero);

printf ("\n\n");

return 0;

}

**Modificación**

\*apEntero = 30;

printf ("\n\nValor de var1: %d", var1);

**Ejemplo 2**

**Estructuras**

#include <stdio.h>

struct complejo {

double real;

double ima;

}n1, n2, res;

int main(){

puts ("introduce numero complejo 1");

scanf("%lf, %lf", &n1.real, &n1.ima);

puts ("introduce numero complejo 2");

scanf("%lf, %lf", &n2.real, &n2.ima);

res.real=n1.real + n2.real;

res.ima=n1.ima + n2.ima;

printf("El resultado es: %lf i + %lf j", res.real, res.ima);

return 0;

}

**Modificación**

struct complejo n1, n2, res;

**Ejemplo 3**

**Typedef**

typedef struct complej complejo;

para el main

complejo n1, n2, res;

**Modificación**

typedef

struct complej{

double real;

double ima;

} complejo;

**Ejemplo 4**

**Funciones**

#include <stdio.h>

#include <math.h>

struct coordenadas

{

float x;

float y;

float z;

};

float distancia(struct coordenadas a, struct coordenadas b);

int main()

{

struct coordenadas punto\_a = { 3.5, 2.5, 1.5 };

struct coordenadas punto\_b = { 5.3, 3.1, 6.3 };

float d;

d = distancia(punto\_a, punto\_b);

printf("%f\n", d);

return 0;

}

float distancia(struct coordenadas a, struct coordenadas b)

{

return sqrtf(pow(a.x - b.x, 2.0) + pow(a.y - b.y, 2.0) + pow(a.z - b.z, 2.0));

}

**Ejemplo 5**

**Manejo de memoria en tiempo de ejecución**

**Malloc**

#include <stdio.h>

#include <stdlib.h>

int main(){

int a,\*b;

a=2;

b=(int\*)malloc(sizeof(int));

\*b=8;

printf("%i \n",\*b);

printf("%i \n",b);

printf("%i \n",a);

free(b);

return 0;

}

**Modificación**

b=(int\*)malloc(sizeof(int));

printf("%i \n",\*b);

free(b);

**Ejemplo 6**

**PILA**

struct Pila{

int tope;

int dato[4];

};

**POP**

int pop(t\_Pila \* pila){

int x=pila->dato[pila->tope];

pila->tope--;

return (x);

}

**PUSH**

void push(t\_Pila \* pila, int valor){

pila->dato[pila->tope]=valor;

pila->tope++;

}

**CREARPILA**

void crearPila(t\_Pila \* pila){

pila->tope=0;

}

**VACIA**

int vacia(t\_Pila \* pila){

if(pila->tope==0)

return 1;

else

return 0;

}

**LLENA**

int llena(t\_Pila \* pila,int tam){

if(pila->tope==tam)

return 1;

else

return 0;

}

**TOPE**

int topeP(t\_Pila \* pila){

return(pila->tope);

}

**VACIAR**

void vaciar(t\_Pila \* pila,int tam){

int i;

for(i=0;i<tam;i++){

pila->tope--;

}

}

**MOSTRAR**

void mostrar(t\_Pila \* pila){

int i;

for(i=0;i<pila->tope;i++){

printf("\nEl elemento %i de la pila es %d\n",i+1,pila->dato[i]);

}

}

**MAIN**

void main(void){

t\_Pila pila;

int num,i=0,op;

crearPila(&pila);

do {

puts("1) Meter un elemento");

puts("2) Sacar un elementos");

puts("3) Tamaño de la Pila");

puts("4) Vaciar la Pila");

puts("5) Mostrar la Pila");

puts("6) Salida");

printf("opcion:\n");

scanf("%i",&op);

switch (op){

case 1:

if(llena(&pila,4)==0){

printf("\nQue dato desea insertar\n");

scanf("%d",&num);

push(&pila, num);

}

else{

printf("\npila llena\n");

}

break;

case 2:

if(vacia(&pila)==0)

printf("\nEl dato eliminado es: %d\n",pop(&pila));

else

printf("\npila vacia\n");

break;

case 3:

printf("\nHay %d datos\n",topeP(&pila));

break;

case 4:

if(vacia(&pila)==0)

vaciar(&pila,topeP(&pila));

else

printf("\npila vacia\n");

break;

case 5:

if(vacia(&pila)==0)

mostrar(&pila);

else

printf("\npila vacia\n");

break;

case 6:

printf("\nAdios...\n");

i=1;

break;

else:

printf("\nOpcion Incorrecta\n");

break;

}

}while(i==0 && i!=1);

}

**Ejemplo 7**

**COLA**

struct Cola{

int inicio;

int final;

int dato[4];

};

**ENCOLAR**

void insertar(t\_Cola \* cola, int valor){

cola->dato[cola->final]=valor;

cola->final ++;

}

**DESENCOLAR**

int remover(t\_Cola \* cola){

int x, i;

x=cola->dato[cola->inicio];

for(i=0;i<cola->final-1;i++){

cola->dato[i]=cola->dato[i+1];

}

cola->final--;

return x;

}

**CREARCOLA**

void crearCola(t\_Cola \* cola){

cola->inicio=0;

cola->final=0;

}

**VACIA**

int vacia(t\_Cola \* cola){

if(cola->final==cola->inicio)

return 1;

else

return 0;

}

**LLENA**

int llena(t\_Cola \* cola,int tam){

if (cola->final==tam)

return 0;

else

return 1;

}

**FINALC**

int finalC(t\_Cola \* cola){

return(cola->final);

}

**VACIAR**

void vaciar(t\_Cola \* cola){

int x, i;

while(cola->inicio!=cola->final){

x=cola->dato[cola->inicio];

for(i=0;i<cola->final-1;i++){

cola->dato[i]=cola->dato[i+1];

}

cola->final--;

}

}

**MOSTRAR**

void mostrar(t\_Cola \* cola){

int i;

for(i=0;i<cola->final;i++){

printf("\nEl elemento %i de la cola es %d\n",i+1,cola->dato[i]);

}

}

**MAIN**

void main(void){

int j=0,num,op,i;

t\_Cola cola;

crearCola(&cola);

do {

puts("1) Meter un elemento");

puts("2) Sacar un elementos");

puts("3) Tamaño de la Cola");

puts("4) Vaciar la Cola");

puts("5) Mostrar la Cola");

puts("6) Salida");

printf("opcion:\n");

scanf("%i",&op);

switch (op){

case 1:

if(llena(&cola,4)==1){

printf("\nQue dato desea insertar\n");

scanf("%d",&num);

insertar(&cola, num);

}

else

printf("\nCola llena\n");

break;

case 2:

if(vacia(&cola)==0)

printf("\nEl dato eliminado es: %d\n",remover(&cola));

else

printf("\nCola vacia\n");

break;

case 3:

printf("\nHay %d datos\n",finalC(&cola));

break;

case 4:

if(vacia(&cola)==0)

vaciar(&cola);

else

printf("\nCola vacia\n");

break;

case 5:

if(vacia(&cola)==0)

mostrar(&cola);

else

printf("\nCola vacia\n");

break;

case 6:

i=1;

break;

default:

printf("\nOpcion Incorrecta\n");

break;

}

}while(i==0);

}

**Ejemplo 8**

**COLA CIRCULAR**

struct ColaC{

int inicio;

int final;

int dato[5];

};

**ENCOLAR**

void insertar(tColaC \* colaC, int valor){

colaC->dato[colaC->final]=valor;

if(colaC->final==5){

colaC->final=1;

}

else{

colaC->final++;

}

if(colaC->inicio==0){

colaC->inicio=1;

}

}

**DESENCOLAR**

int remover(tColaC \* colaC){

int x;

x=colaC->dato[colaC->inicio];

if(colaC->inicio==5){

colaC->inicio=1;

}

else{

colaC->inicio++;

}

if(colaC->inicio==colaC->final){

colaC->inicio=0;

colaC->final=0;

}

return x;

}

**CREACOLA**

void crearCola(tColaC \* colaC){

colaC->inicio=0;

colaC->final=0;

}

**VACIA**

int vacia(tColaC \* colaC){

if(colaC->inicio==0 && colaC->final==0)

return 1;

else

return 0;

}

**LLENA**

int llena(tColaC \* colaC){

if (colaC->final==5 && colaC->inicio==1)

return 0;

else

return 1;

}

**MAIN**

void main(void){

int j=0,num,op,i;

tColaC colaC;

crearCola(&colaC);

do {

puts("1) Meter un elemento");

puts("2) Sacar un elementos");

puts("3) Salida");

printf("opcion:\n");

scanf("%i",&op);

switch (op){

case 1:

if(llena(&colaC)==1){

printf("\nQue dato desea insertar\n");

scanf("%d",&num);

insertar(&colaC, num);

}

else{

printf("Cola llena");

}

break;

case 2:

if(vacia(&colaC)==0){

printf("\nEl dato eliminado es: %d\n",remover(&colaC));

}

else{

printf("Cola vacia");

}

break;

case 3:

i=1;

printf("\n\tHasta luego");

break;

default:

printf("\nOpcion Incorrecta\n");

break;

}

}while(i==0);

}

**Ejemplo 8**

**BICOLA CIRCULAR**

struct ColaCD{

int inicio;

int final;

int dato[5];

};

**ENCOLARFINAL**

void insertarFin(tColaCD \* colaCD, int valor){

colaCD->dato[colaCD->final]=valor;

if(colaCD->final==5){

colaCD->final=1;

}

else{

colaCD->final++;

}

if(colaCD->inicio==0){

colaCD->inicio=1;

}

}

**DESENCOLARINICIO**

int removerIni(tColaCD \* colaCD){

int x;

x=colaCD->dato[colaCD->inicio];

if(colaCD->inicio==5){

colaCD->inicio=1;

}

else{

colaCD->inicio++;

}

if(colaCD->inicio==colaCD->final){

colaCD->inicio=0;

colaCD->final=0;

}

return x;

}

**DESENCOLARFINAL**

int removerFin(tColaCD \* colaCD){

int x;

x=colaCD->dato[colaCD->final];

if(colaCD->final==5){

colaCD->final=1;

}

else{

colaCD->final++;

}

if(colaCD->final==colaCD->inicio){

colaCD->inicio=0;

colaCD->final=0;

}

return x;

}

**ENCOLARINICIO**

void insertarIni(tColaCD \* colaCD, int valor){

colaCD->dato[colaCD->inicio]=valor;

if(colaCD->inicio==5){

colaCD->inicio=1;

}

else{

colaCD->inicio++;

}

if(colaCD->final==0){

colaCD->final=1;

}

}

**CREACOLA**

void crearCD(tColaCD \* colaCD){

colaCD->inicio=0;

colaCD->final=0;

}

**VACIA**

int vacia(tColaCD \* colaCD){

if(colaCD->inicio==0 && colaCD->final==0)

return 1;

else

return 0;

}

**LLENA**

int llena(tColaCD \* colaCD){

if (colaCD->final==5 && colaCD->inicio==1)

return 0;

else

return 1;

}

**MAIN**

void main(void){

int j=0,num,op,i;

tColaCD colaCD;

crearCD(&colaCD);

do {

puts("1) Meter un elemento al final");

puts("2) Sacar un elemento del inicio");

puts("3) Meter un elemento al inicio");

puts("4) Sacar un elemento del final");

printf("opcion:\n");

scanf("%i",&op);

switch (op){

case 1:

if(llena(&colaCD)==1){

printf("\nQue dato desea insertar\n");

scanf("%d",&num);

insertarFin(&colaCD, num);

}

else{

printf("Cola llena");

}

break;

case 2:

if(vacia(&colaCD)==0){

printf("\nEl dato eliminado es: %d\n",removerIni(&colaCD));

}

else{

printf("Cola vacia");

}

break;

case 3:

if(llena(&colaCD)==1){

printf("\nQue dato desea insertar\n");

scanf("%d",&num);

insertarIni(&colaCD, num);

}

else{

printf("Cola llena");

}

break;

case 4:

if(vacia(&colaCD)==0){

printf("\nEl dato eliminado es: %d\n",removerFin(&colaCD));

}

else{

printf("Cola vacia");

}

break;

default:

printf("\nOpcion Incorrecta\n");

i=1;

break;

}

}while(i==0);

}