/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Online C Compiler.

Code, Compile, Run and Debug C program online.

Write your code in this editor and press "Run" button to compile and execute it.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include "myfile.h"

//\*\*\*\*\*\*\*\* Variable desclaration\*\*\*\*\*\*\*\*\*//

int Total, sum1, sum2;

extern int support; // Here, name of global variable should be same as declared in header file like myfile.h

int main()

{

int loop;

for(loop = 0; loop <1; loop++)

{

//Fn\_precedence();

//Fn\_do\_loop();

//Fn\_Pattern\_minus(); // For generating pattern.

//Fn\_Pattern\_plus(); // For generating pattern.

//Fn\_CallbyValue(); // Original Value is not modified.

//Fn\_CallbyReference(); // Original Value is modified.

//Fn\_recursion(); // Functiona called itself.

//Fn\_Array\_init();

//Fn\_ArrayPass();

Fn\_Pointer();

}

return 0;

}

void Fn\_precedence()

{

int Vr\_precedence;

Vr\_precedence = (2\*8)+4\*2+44;

printf("Precedence: %d\n\"\?", Vr\_precedence);

}

void Fn\_do\_loop()

{

int Vr\_do\_inc = 0;

do

{

Vr\_do\_inc += 1;

printf("Vr\_do\_inc: %d\n", Vr\_do\_inc);

} while (Vr\_do\_inc < 50);

}

void Fn\_Pattern\_minus()

{

int Vr\_innerlp=5, Vr\_outerlp=0,Vr\_print=1;

for(Vr\_outerlp; Vr\_outerlp <=4; Vr\_outerlp++)

{

for(Vr\_innerlp = 4; Vr\_innerlp >= Vr\_outerlp; Vr\_innerlp--)

{

printf("%d",Vr\_print);

Vr\_print += 1;

}

printf("\n");

Vr\_print = 1;

}

}

void Fn\_Pattern\_plus()

{

int Vr\_innerlp=0, Vr\_outerlp=0,Vr\_print=1;

for(Vr\_outerlp; Vr\_outerlp <=4; Vr\_outerlp++)

{

for(Vr\_innerlp = 0; Vr\_innerlp <= Vr\_outerlp; Vr\_innerlp++)

{

printf("%d",Vr\_print);

Vr\_print += 1;

}

printf("\n");

Vr\_print = 1;

}

}

void Fn\_Test\_CallbyValue(int num)

{

printf("Before adding value inside the function: %d\n", num);

num = num + 100;

printf("After adding value inside the function: %d\n", num);

}

void Fn\_CallbyValue()

{

int Vr\_value = 100;

printf("Before function calling: %d\n", Vr\_value);

Fn\_Test\_CallbyValue(Vr\_value);

printf("After function calling: %d\n", Vr\_value);

}

void Fn\_Test\_CallbyReference(int \*num)

{

printf("Before adding value inside the function: %d\n", \*num);

\*num = \*num + 100;

printf("After adding value inside the function: %d\n", \*num);

}

void Fn\_CallbyReference()

{

int Vr\_value = 100;

printf("Before function calling: %d\n", Vr\_value);

Fn\_Test\_CallbyReference(&Vr\_value);

printf("After function calling: %d\n", Vr\_value);

}

int factorial(int number)

{

if (number == 0)

return 1;

return (number \* factorial(number - 1));

}

void Fn\_recursion()

{

int fact = 0, Vr\_fact\_num;

printf("Enter the factorail number: ");

scanf("%d", &Vr\_fact\_num);

fact = factorial(Vr\_fact\_num);

printf("\nFactorial of %d is: %d",Vr\_fact\_num,fact);

}

void Fn\_Array\_init()

{

int Ar\_Subject\_Marks[5] = {45,47,48,49,50}, Vr\_Subjects = 5, Lp\_Sum,Vr\_Total\_Sum = 0;

float Vr\_Avg;

for (Lp\_Sum=0; Lp\_Sum <=4;Lp\_Sum ++)

{

Vr\_Total\_Sum = Vr\_Total\_Sum + Ar\_Subject\_Marks[Lp\_Sum];

printf("Total Sum of Marks: %d\n",Vr\_Total\_Sum);

}

Vr\_Avg = Vr\_Total\_Sum / Vr\_Subjects;

printf("Avg Marks: %f\n", Vr\_Avg);

}

int minarray(int arr[], int size)

{

int min = arr[0], i=0;

printf("min: %d\n", min);

for(i=0; i<size;i++)

{

if(min > arr[i])

{

min = arr[i];

}

}

return min;

}

void Fn\_ArrayPass()

{

int Array\_numbers[]={4,5,7,3,8,0,9}, min = 0;

min = minarray(Array\_numbers,7);

printf("Minimum number: %d", min);

}

void Fn\_Pointer()

{

int number=50;

int \*p;//pointer to int

int \*\*p2;//pointer to pointer

p=&number;//stores the address of number variable

p2=&p;

printf("Address of number variable is %x \n",&number);

printf("Address of p variable is %x \n",p);

printf("Value of \*p variable is %d \n",\*p);

printf("Address of p2 variable is %x \n",p2);

printf("Value of \*\*p2 variable is %d \n",\*\*p2);

}