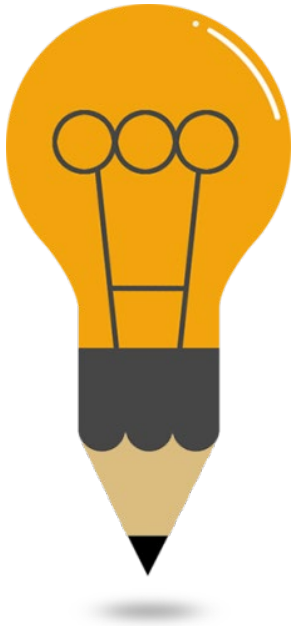


STRUKTUR DATA

Pertemuan 2



Agenda Pertemuan



1

Pengenalan Bahasa C

2

Tipe Data dalam Bahasa C

3

Array

Pengenalan Bahasa C

- **C** adalah bahasa pemrograman tingkat menengah. Dapat digunakan untuk pemrograman tingkat rendah (*assembly language*) seperti kernel, driver, dll dan juga mendukung fungsi – fungsi dalam bahasa pemrograman tingkat atas (Java, Python, Ruby, dll)
- **C** adalah bahasa pemrograman terstruktur (program dapat dipecah menjadi beberapa program yang lebih kecil – ***function***) . Contoh: program untuk mengelola nilai mahasiswa

```
main() Nilai Mahasiswa
      HitungNilai()
      TampilkanNilai()
      InputNilai()
      UpdateNilai()
      GetNilaiMahasiswaByNIM()
```

- **C** adalah *case-sensitive* (huruf kecil dan huruf besar berbeda/berpengaruh)
- **C** adalah *general purpose programming language* (bertujuan umum) untuk menyelesaikan banyak masalah, *tidak spesifik* masalah tertentu.



Pengenalan Bahasa C

```
#include <stdio.h>

int main() {
    /* my first program in C */
    printf("Hello, World! \n");

    return 0;
}
```

- Ekstensi file “.c”
- `include stdio.h` = header file (library) pada bahasa C yang digunakan untuk operasi input-output (`stdio` = Standar Input dan Output). Tanpa menggunakan library ini maka perintah-perintah input dan output tidak dapat dieksekusi. Fungsi – fungsi di `stdio.h` meliputi:

Library Functions:

<code>clearr()</code>	<code>fclose()</code>	<code>fccloseall()</code>	<code>fdopen()</code>	<code>fflush()</code>
<code>fgetc()</code>	<code>fgetchar()</code>	<code>fgetpos()</code>	<code>fgets()</code>	<code>flushall()</code>
<code>fopen()</code>	<code>fprint()</code>	<code>fputc()</code>	<code>fputchar()</code>	<code>fputs()</code>
<code>fread()</code>	<code>free()</code>	<code>freopen()</code>	<code>fscan()</code>	<code>fseek()</code>
<code>fsetpos()</code>	<code>ftell()</code>	<code>fwrite()</code>	<code>gets()</code>	<code>getw()</code>
<code>perror()</code>	<code>printf()</code>	<code>puts()</code>	<code>rename()</code>	<code>rewind()</code>
<code>scanf()</code>	<code>unlink()</code>			

Pengenalan Bahasa C

Contoh beberapa header file untuk program C:

- **stddef.h** – mendefinisikan makro
- **stdint.h** – mendefinisikan jenis integer dengan lebar tertentu
- **stdio.h** – mendefinisikan fungsi input-output standar
- **stdlib.h** – mendefinisikan fungsi konversi numerik dan alokasi memori
- **string.h** – mendefinisikan fungsi untuk penanganan string
- **math.h** – mendefinisikan fungsi matematika yang umum

Pengenalan Bahasa C

```
#include <stdio.h>

int main() {
    /* my first program in C */
    printf("Hello, World! \n");

    return 0;
}
```

- `int main()` Fungsi utama dimana program mulai dieksekusi
- `/* ...*/` atau `//` adalah *comments* dan akan diacuhkan oleh compiler
- `printf()` adalah fungsi yang akan menampilkan kata – kata “Hello, World!”
- `return 0` akan mengakhiri fungsi `main()`
- Setiap statement diakhiri dengan titik koma (;)

Contoh:

```
// menuliskan kalimat
#include <stdio.h>

int main() {
    printf("Selamat, Anda telah berhasil\n");
    printf("membuat program C yang pertama!\n");
    return 0;
}
```

Komentor, tidak diproses header

Pindah baris

Akhir perintah

Perintah untuk tampilkan ke layar

Pengenalan Bahasa C

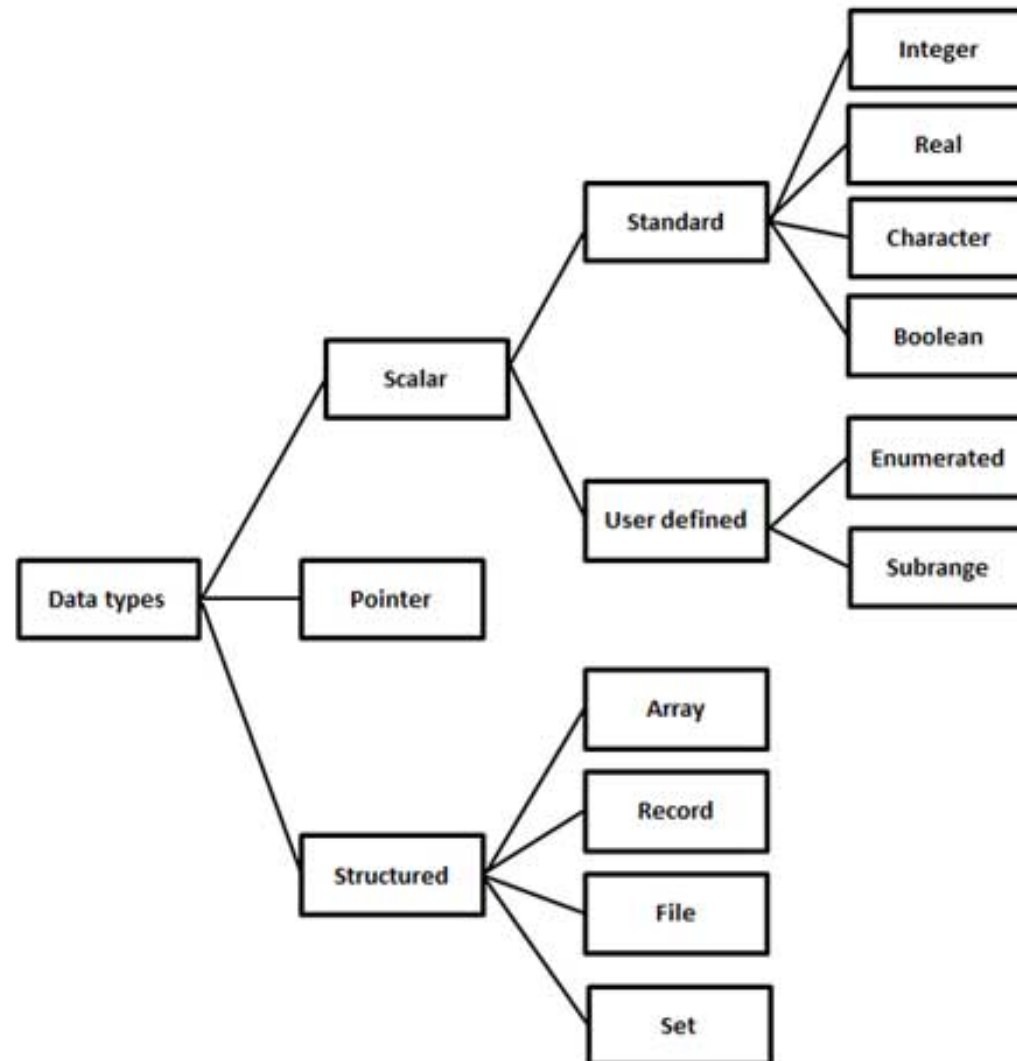
■ **Identifier** : untuk memberi nama/mengidentifikasi variabel, fungsi, dll

■ A - Z, a - z, *underscore*, dan digit 0 – 9

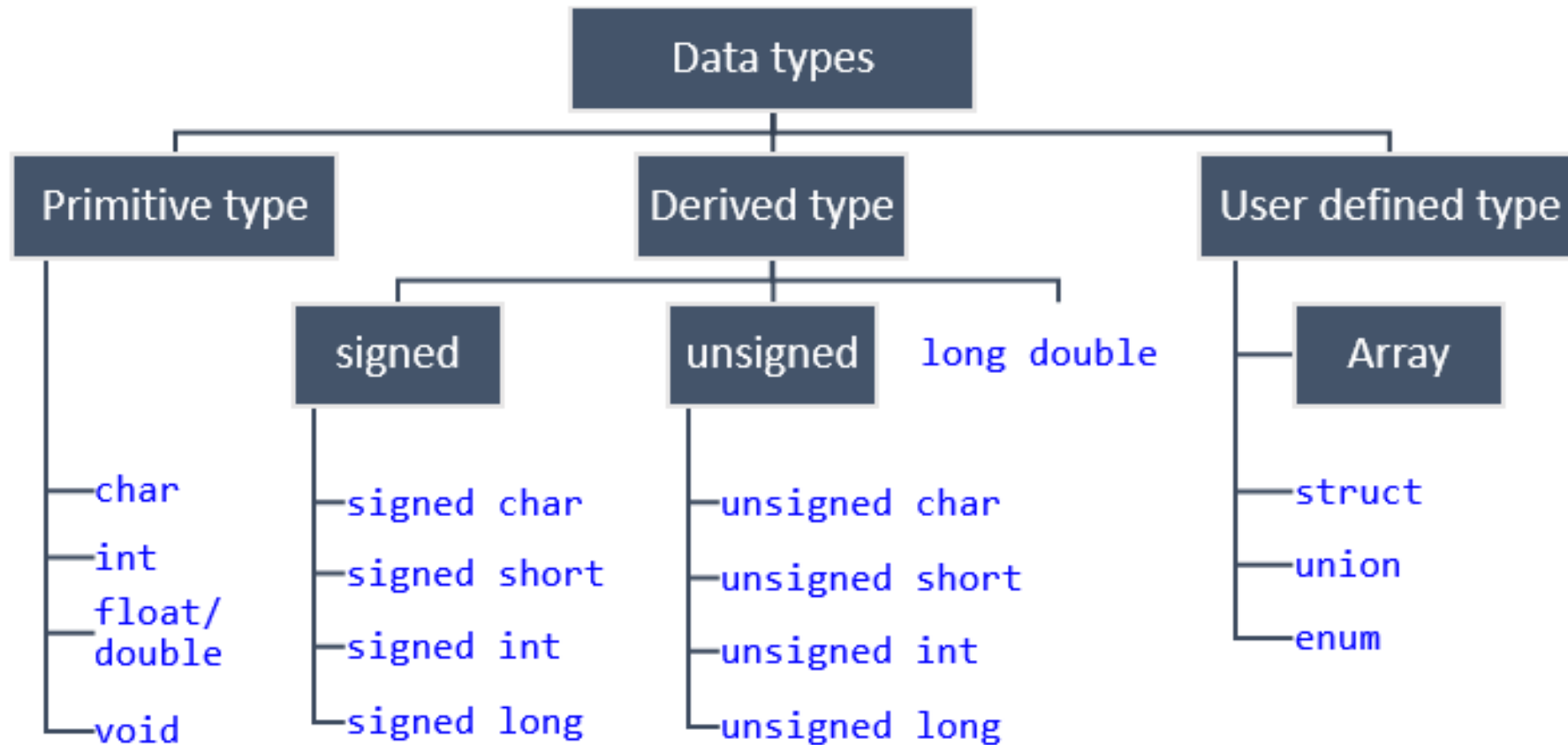
■ **Selain reserved words** berikut:

auto	else	long	switch
break	enum	register	typedef
case	extern	return	union
char	float	short	unsigned
const	for	signed	void
continue	goto	sizeof	volatile
default	if	static	while
do	int	struct	_Packed
double			

Tipe Data dalam Pascal



Tipe Data dalam Bahasa C



Tipe Data dalam Bahasa C

1. Tipe data primitive (tipe data sederhana)

- hanya mampu menyimpan satu nilai pada setiap satu variabel.
- mempunyai besaran memori yang tetap dan pasti

- Pascal:

Contoh jangkauan (range) variabel:

Tipe Data Pascal	Karakteristik	Contoh
String	Teks	'New York', 'My Name'
Integer	Bilangan bulat	23, 16595, 0, -632
Real	Bilangan desimal	3.14, 503.2
Boolean	True atau False	TRUE, FALSE
Character	Satu karakter	A, e, u, 9, o, 3

Tipe Data	Minimum	Maximum
Integer	-32,768	32,767
LongInt	-2,147,483,648	2,147,487,647
ShortInt	-128	127
Real	2.9 x 10e-39	1.7 x 10e+38

Tipe Data dalam Bahasa C

■ Bahasa C:

Tipe Data C	Range	Kode Format
char	-128 to 127 atau 0 to 255	%c
int	-32,768 to 32,767 atau -2,147,483,648 to 2,147,483,647	%d
float	3.4e-038 to 3.4e+038 (6 angka di belakang koma)	%f
double	2.3e-308 to 1.7e+308 (15 angka di belakang koma)	%f
void	(tidak bertipe, tidak menyimpan apapun, biasanya untuk tipe fungsi yang tidak return value apapun)	

*Ukuran dan range dari setiap tipe data tergantung mesin (misal mesin 32bit bisa memberikan hasil yang berbeda dari 64bit) dan variasi compiler

Tipe Data dalam Bahasa C

2. Tipe data turunan (*derived*)

- Kombinasi *qualifiers* dan tipe data primitive

```
[sign-qualifier] [size-qualifier] <basic-data-type>
```

- [...] : optional (boleh ada atau tidak)
- Contoh: `unsigned short int`
- **Size qualifier** digunakan untuk mengubah ukuran : `short` atau `long`
- **Sign qualifier** digunakan untuk menentukan apakah variabel tersebut dapat menampung nilai negative atau tidak: `signed` (bisa +/-) atau `unsigned` (hanya +)

Tipe Data dalam Bahasa C

2. Tipe data turunan (*derived*)

■ Contoh:

Tipe Data	Ukuran (byte)	Range
int (atau signed int)	2 atau 4*	-32,768 to 32,767 atau -2,147,483,648 to 2,147,483,647
unsigned int	2 atau 4*	0 to 65,535 atau 0 to 4,294,967,295
long int (biasa ditulis long saja)	8	-2,147,483,648 to 2,147,483,647
double	8	2.3e-308 to 1.7e+308
long double	10	3.4e-4932 to 1.1e+4932

**tergantung prosesor (32bit / 64bit)*


Tipe Data dalam Bahasa C

■ Deklarasi di bahasa C:

```
#include <stdio.h>
int main() {
    int a = -128;
    char ch2 = 'a';
    unsigned char uc = 'b';
    short s = 10;
    int i = 1000;
    unsigned int ui = 45555;
    long l = 1234567;
    unsigned long ul = 1234567898;
    float f = 3.5;
    double d = 23.9999;
    long double ld = 23.239;

    printf("Nilai dari variabel adalah : %d, %c, dan %f", a, ch2, d);

    return 0;
}
```



Format output
untuk integer,
char, dan
double/float

■ Ukuran dan range tergantung mesin dan compiler

```
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
#include <float.h>

int main(int argc, char** argv) {

    printf("CHAR_BIT      :   %d\n", CHAR_BIT);
    printf("CHAR_MAX      :   %d\n", CHAR_MAX);
    printf("CHAR_MIN      :   %d\n", CHAR_MIN);
    printf("INT_MAX       :   %d\n", INT_MAX);
    printf("INT_MIN       :   %d\n", INT_MIN);
    printf("LONG_MAX      :   %ld\n", (long) LONG_MAX);
    printf("LONG_MIN      :   %ld\n", (long) LONG_MIN);
    printf("SCHAR_MAX     :   %d\n", SCHAR_MAX);
    printf("SCHAR_MIN     :   %d\n", SCHAR_MIN);
    printf("SHRT_MAX      :   %d\n", SHRT_MAX);
    printf("SHRT_MIN      :   %d\n", SHRT_MIN);
    printf("UCHAR_MAX     :   %d\n", UCHAR_MAX);
    printf("UINT_MAX      :   %u\n", (unsigned int) UINT_MAX);
    printf("ULONG_MAX     :   %lu\n", (unsigned long) ULONG_MAX);
    printf("USHRT_MAX     :   %d\n", (unsigned short) USHRT_MAX);

    return 0;
}
```

Output:

CHAR_MAX : 127 (range maximum)

CHAR_MIN : -128 (range minimum)

Tipe Data dalam Bahasa C

3. Tipe data yang didefinisikan user (*user-defined*)

- **Array:** Tipe data yang terdiri dari kumpulan tipe data dasar. Tipe data tersebut harus 1 jenis.
- **Pointer:** Tipe data untuk mengakses alamat memory suatu variabel secara langsung.
- **Structure:** Tipe data yang terdiri dari kumpulan tipe data dasar. Tipe data tersebut bisa lebih dari 1 jenis.
- **Union:** Tipe data yang menyimpan beberapa tipe data berbeda dalam lokasi memory yang sama.

Array

■ Format deklarasi dalam bahasa C: `tipe_data nama_variabel [jumlah_element]`

■ Deklarasi Array 1 dimensi

■ `float bilangan[100]`

■ `char huruf[3]`

■ Mengisi nilai Array:

```
int bilangan[5];
```

Atau

```
int bilangan[5] = {6, 9, -8, 24, -99};
```

```
bilangan[0] = 6;  
bilangan[1] = 9;  
bilangan[2] = -8;  
bilangan[3] = 24;  
bilangan[4] = -99;
```

Contoh Deklarasi dan Mengisi Nilai Array

```
#include <stdio.h>

int main(void)
{
    int bilangan[5];

    bilangan[0] = 6;
    bilangan[1] = 9;
    bilangan[2] = -8;
    bilangan[3] = 24;
    bilangan[4] = -99;

    printf("Isi array bilangan pertama: %d \n",bilangan[0]);
    printf("Isi array bilangan kedua: %d \n",bilangan[1]);
    printf("Isi array bilangan ketiga: %d \n",bilangan[2]);
    printf("Isi array bilangan keempat: %d \n",bilangan[3]);
    printf("Isi array bilangan kelima: %d \n",bilangan[4]);

    return 0;
}
```

Deklarasi Array
tanpa
mendefinisikan
jumlah elemen

Atau

```
#include <stdio.h>

int main(void)
{
    int bilangan[5] = {6, 9, -8, 24, -99};

    printf("Isi array bilangan pertama: %d \n",bilangan[0]);
    printf("Isi array bilangan kedua: %d \n",bilangan[1]);
    printf("Isi array bilangan ketiga: %d \n",bilangan[2]);
    printf("Isi array bilangan keempat: %d \n",bilangan[3]);
    printf("Isi array bilangan kelima: %d \n",bilangan[4]);

    return 0;
}
```

```
#include <stdio.h>

int main(void)
{
    char kumpulan_huruf[] = {'a', 'C', 'x'};

    printf("Isi array kumpulan_huruf: ");
    printf("%c, %c, %c \n",kumpulan_huruf[0],kumpulan_huruf[1],kumpulan_huruf[2]);

    return 0;
}
```

Contoh Mengubah Nilai Elemen Array

```
#include <stdio.h>

int main(void)
{
    float pecahan[] = {3.14, -99.01, 0.002};

    printf("Isi array pecahan: ");
    printf("%.3f, %.3f, %.3f \n", pecahan[0], pecahan[1], pecahan[2]);
    printf(" \n");

    pecahan[1] = 9.123;
    pecahan[2] = 12.9925;

    printf("Isi array pecahan: ");
    printf("%.3f, %.3f, %.3f \n", pecahan[0], pecahan[1], pecahan[2]);

    return 0;
}
```

Array 2 Dimensi

- Array 2 dimensi: sebutan untuk array yang penomoran index-nya menggunakan 2 buah angka

```
#include <stdio.h>

int main(void)
{
    int bilangan[2][2];

    bilangan[0][0] = 100;
    bilangan[0][1] = 101;
    bilangan[1][0] = 110;
    bilangan[1][1] = 111;

    printf("Isi array bilangan: \n");
    printf("%d, %d \n", bilangan[0][0], bilangan[0][1]);
    printf("%d, %d \n", bilangan[1][0], bilangan[1][1]);

    return 0;
}
```

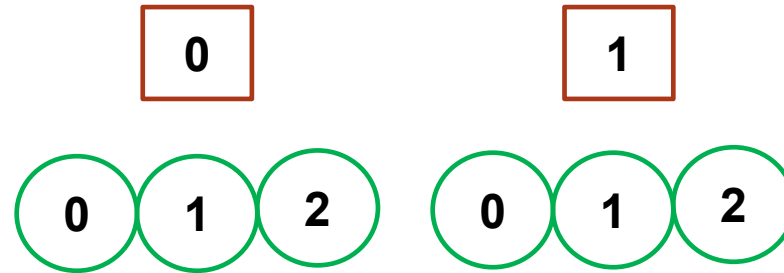
```
#include <stdio.h>

int main(void)
{
    int matrix[2][3] = {{1,2,3},{7,8,9}};

    printf("Isi array matrix: \n");
    printf("%d %d %d \n", matrix[0][0], matrix[0][1], matrix[0][2]);
    printf("%d %d %d \n", matrix[1][0], matrix[1][1], matrix[1][2]);

    return 0;
}
```

Index Array Multi Dimensi



```
int matrix[2][3] = {{1, 2, 3}, {7, 8, 9}};
```

Berapakah nilai `matrix[1][0]` ?

Array 3 Dimensi dan String (Array of char)

■ Array 3 dimensi

```
#include <stdio.h>

int main(void)
{
    int matrix[2][3][4] =
    {
        { {7, 4, 12, 3}, {-9, 29, 3, 11}, {6, 34, 23, 20} },
        { {6, 15, 1, 5}, {17, 8, -3, 15}, {99, -1, 44, 9} }
    };

    printf("Isi matrix[0][0][0]: %d \n",matrix[0][0][0]);
    printf("Isi matrix[0][1][0]: %d \n",matrix[0][1][0]);
    printf("Isi matrix[1][1][3]: %d \n",matrix[1][1][3]);
    printf("Isi matrix[1][2][3]: %d \n",matrix[1][2][3]);

    return 0;
}
```

■ String

```
char c[] = "abcd";

char c[50] = "abcd";

char c[] = {'a', 'b', 'c', 'd', '\0'};

char c[5] = {'a', 'b', 'c', 'd', '\0'};
```

```
#include <stdio.h>
int main()
{
    char name[20];
    printf("Enter name: ");
    scanf("%s", name);
    printf("Your name is %s.", name);
    return 0;
}
```

Format
output untuk
string

Latihan Soal (PR)

1. Translasikan (terjemahkan) solusi birthday cake candles (pertemuan sebelumnya) yang dijelaskan dalam bahasa Pascal ke C!

■ If...else if...else statement dalam bahasa C

```
if(boolean_expression 1) {  
    /* Executes when the boolean expression 1 is true */  
} else if( boolean_expression 2) {  
    /* Executes when the boolean expression 2 is true */  
} else if( boolean_expression 3) {  
    /* Executes when the boolean expression 3 is true */  
} else {  
    /* executes when the none of the above condition is true */  
}
```

```
(A == B)  
(A != B)  
(A > B)  
(A < B)  
(A >= B)  
(A <= B)
```

■ Looping (for) statement dalam bahasa C

```
for (initializationStatement; testExpression; updateStatement)  
{  
    // statements inside the body of loop  
}
```

```
int i;  
for (i = 1; i < 11; i++)  
{  
    printf("%d ", i);  
}
```