BITS Pilani

Hyderabad Campus

CS F111: Computer Programming

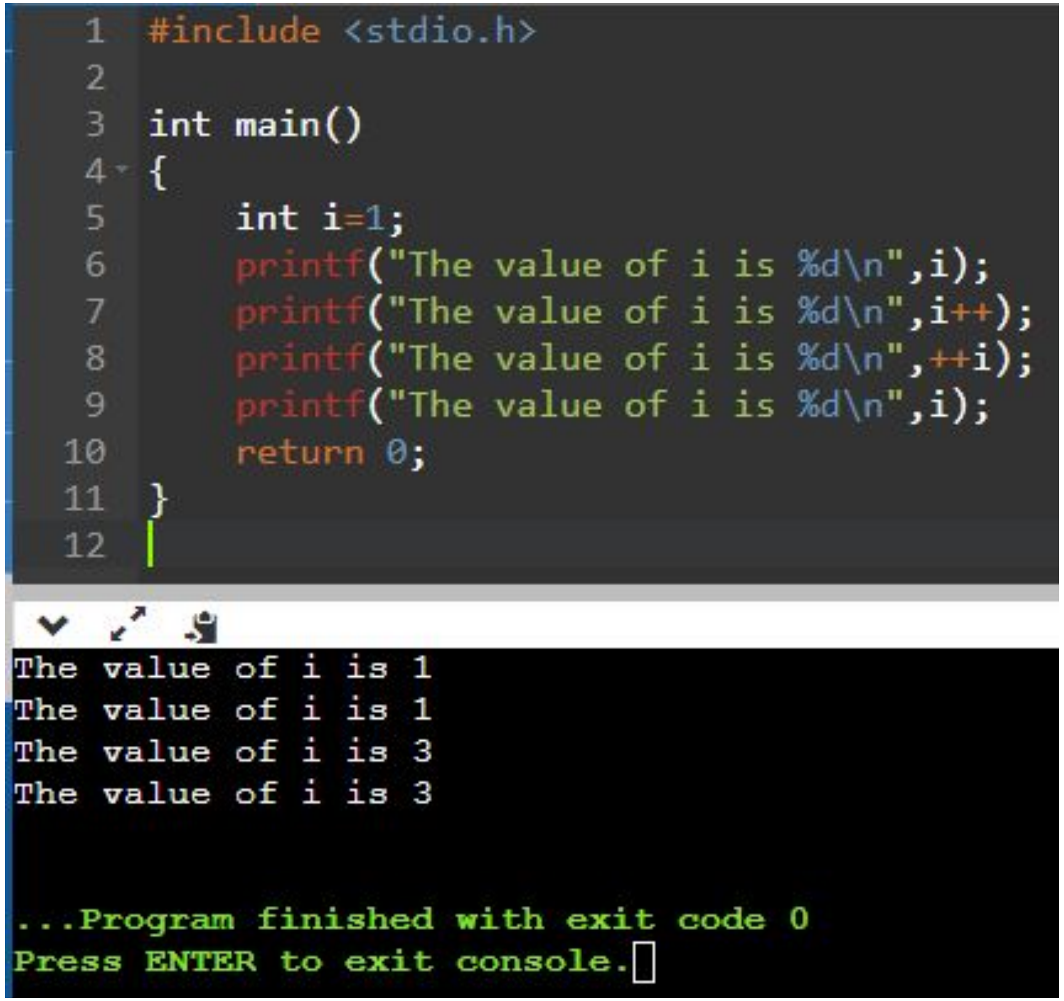
(Second Semester 2020-21)

Lect 9: Special Operators and Type Casting

Dr. Nikumani Choudhury
Asst. Prof., Dept. of Computer Sc. & Information Systems
nikumani@hyderabad.bits-pilani.ac.in

Recap: Increment Operator

```
#include <stdio.h>
int main()
{
        int i=1;
        printf("The value of i is %d\n",i);
        printf("The value of i is %d\n",i++);
        printf("The value of i is %d\n",++i);
        printf("The value of i is %d\n",i);
        return 0;
}
```



```
#include <stdio.h>

int main()
{
    int i=1;
    printf("The value of i is %d\n",i);
    printf("The value of i is %d\n",i++);
    printf("The value of i is %d\n",++i);
    printf("The value of i is %d\n",i);
    return 0;
}

```

```
The value of i is 1
The value of i is 1
The value of i is 3
The value of i is 3


...Program finished with exit code 0
Press ENTER to exit console.
```

Special Operator: Comma

```
#include <stdio.h>
int main ()
{
  int i = (5,10);
  printf("%d", i);
  return 0;
}
```

```
int j = (f1(), f2());
```

```
#include <stdio.h>
int main ()
{
  int x = 10, y;
  y = (x++, printf ("x =
  %d\n", x), ++x, printf("x =
  %d\n", x), x++);
  printf("y = %d\n", y);
  printf("x = %d\n", x);
  return 0;
}
```

Output?

Special Operators: Dereference and Address

- Dereference or Indirection operator *
- Address operator &

```
#include <stdio.h>
int main()
{
 int x = 10;
 int *p;
 p = &x;
 printf ("value of x: %d\n",*p);
 return 0;
}
```

Two more we will discuss later: ->, and .

Let us see the run on onlinegdb…

Real Arithmetic

- Arithmetic operations involving only real or floating-point operands.

- Since floating-point values are rounded to the number of significant digits permissible, the final value is an approximation of the final result.

- ***The modulus operator cannot be used with real operands.***

Mixed-mode Arithmetic

- **When one of the operands is integer and the other is real, the expression is called a *mixed-mode* arithmetic expression.**

- **If either operand is of the real type, then only real arithmetic is performed, and the result is a real number.**

25 / 10 □ 2

25 / 10.0 □ 2.5

25.0 / 10 □ 2.5

Typecasting

- Implicit :

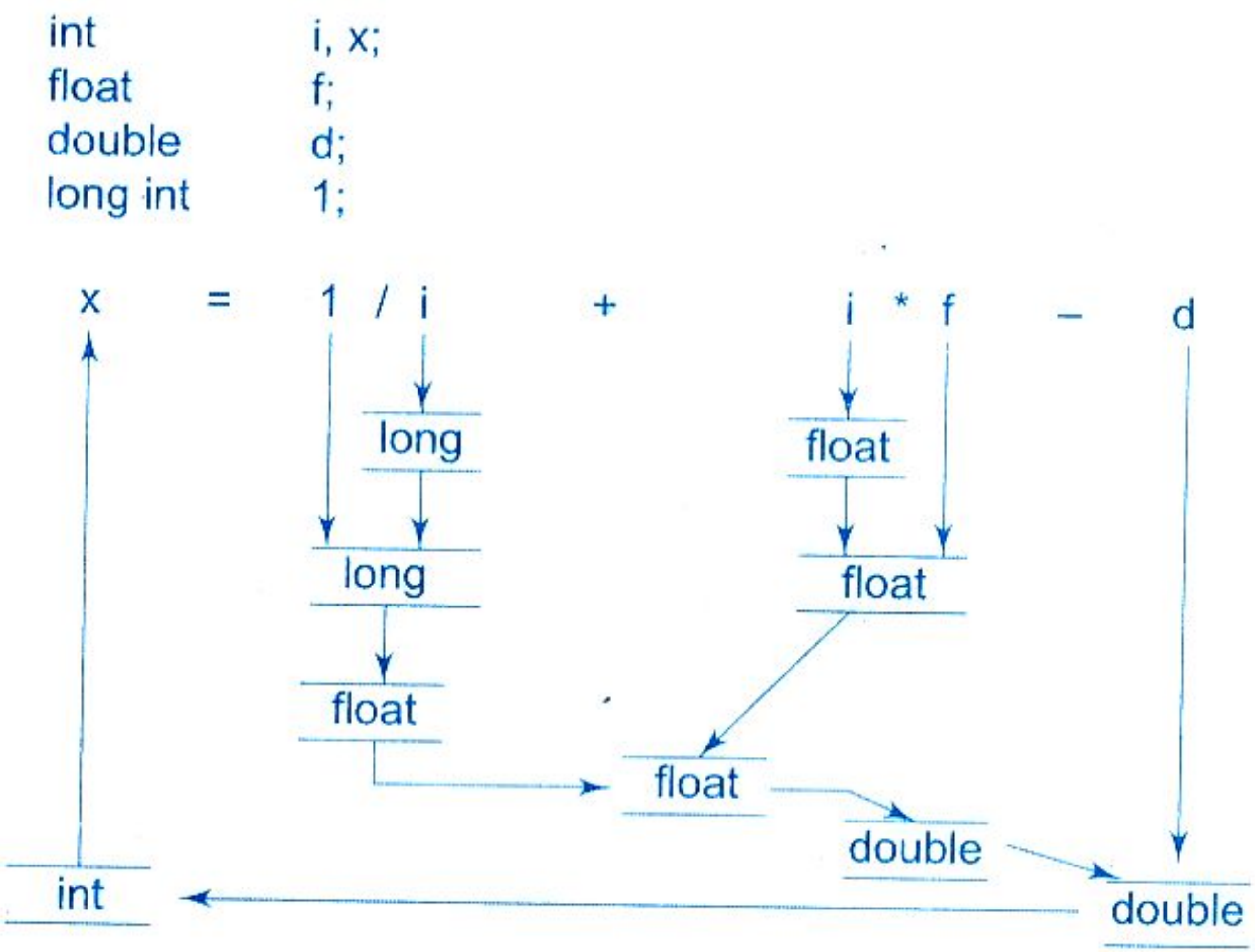
```
int a = 25;
double b = a + 7.8;
```

- Explicit : (data-type) a;

```
 int a = 25, b = 10;
double x = (double)a / b;  // x = 2.5
```

so here for this operation a get’s promoted to double

Type Conversions: Implicit



```
int          i, x;
float        f;
double       d;
long int     1;
```

```
#include <stdio.h>
int main()
{
  int j;
  double i = 23.8;
   j=i+8;
   printf ("%d\n",j);
  return 0;
}
```

31

Type Conversions: Explicit

```
main()
  {
     float  sum ;
     int    n ;

     sum = 0 ;

     for( n = 1 ; n <= 10 ; ++n )
     {
       sum = sum + 1/(float)n ;
       printf("%2d  %6.4f\n", n, sum) ;
     }
  }
```

Output

```
 1  1.0000
 2  1.5000
 3  1.8333
 4  2.0833
 5  2.2833
 6  2.4500
 7  2.5929
 8  2.7179
 9  2.8290
10  2.9290
```

```
#include<stdio.h>

int main()
{
   double x = 1.2;

   int sum = (int)x + 1;

   printf("sum = %d", sum);

   return 0;
}
```

Output:

```
sum=2
```

Type Casting:

It helps us to compute expressions containing variables of different data types.