COMP133: **COMPUTER AND PROGRAMMING**

Overview of C-programming

Dr. Radi Jarrar
Department of Computer Science
Birzeit University

What is C-language?

- C is a general purpose procedural programming language.
- Developed in 1972 and considered one of the strongest programming languages.
- Many programming languages are borrowed (directly or indirectly) from C such as C++, C#, Java, PHP, Perl, Python.
- C is closely tied to the development of the UNIX operating system.

Compiler

- The C-program is translated to machine language using compilers.
- A compiler is a special program that processes statements of a specific programming language and “translates” them into machine language that the processor can use.

Example C-program

```
#include <stdio.h>

int main(void)
{
    printf("hello, world\n");

    return 0;
}
```

Example C-program

```
//C program for area of circle Comment
#include <stdio.h> // standard header file (contains printf and
scanf )
#define PI  3.141   //we use define for creating constant
int main()  //int, float , and return   (reserved words)
{
      float r, a;   // r, a are variables
      printf("Please enter the radius: ");  //standard identifier
      scanf("%f", &r);      //standard identifier
      a = PI * r * r;       //  = , *,{, }  special symbols
      printf("%f\n", a);    //standard identifier
      return 0;
}
```

Preprocessor Directives

- `#include`
  - gives a program access to a library
- `<stdio.h>`
  - standard header file
  - contains information about standard input and output functions such as `scanf` and `printf`

Preprocessor Directives

- `#include <stdio.h>`
  - notify the preprocessor that some names used in the program are found in `<stdio.h>`
- `#define`
  - using only data values that never change should be given names

Preprocessor Directives

- Constant macro
  - a name that is replaced by a particular constant value
- E.g.,:
  - `#define PI  3.141593`
  - `#define MAX_LENGTH  100`

Comments

- Two types:
  - One-line comment //
  - Multiple-line comment /* */
- Examples:
  - //This is a one-line comment
  - /*Hello, this is
  - multiple-line comment*/

Variable declarations and data types

- Variable: a name associated with a memory cell whose value can change.
- Examples:
- sum , x ,y , result,…..

Variable declarations and data types

1. A variable must consist only of letters, digits, and underscores.
2. A variable cannot begin with a digit.
3. A C reserved word cannot be used as a user variable.
4. A variable defined in a C standard library should not be redefined.

- Reserved words : A word that has special meaning in C
- for example: int, float, double, char , return ,…, etc.

Variable declarations and data types

- Syntax :
  - int *variable_list;*
  - float *variable_list;*
  - char *variable_list;*
- Examples :
  - `int count, large;`
  - `Float ans;  or  float ans=4.2;`
  - `char first_initial;`

Variable declarations and data types

- **Data types:**
- int (16 bit)
- float (32 bit)
- double (64 bit)
- char (1-byte)
- a real number has an integral part and a fractional part that are separated by a decimal point

Variable declarations and data types

- **Data types:**
- char (8 bit)
  - represent an individual character value
  - include a letter, adigit, a special symbol
  - E.g., `'A', 'z', '2', '9', '*', ':', '"', ' '`

Variable declarations and data types

Invalid variables names

Invalid identifier	Reason Invalid
1Letter	begins with a digit
double	reserved word
int	reserved word
TWO*FOUR	character * not allowed
joe's	character ' not allowed

Variable declarations and data types

Reserved Words	Standard Identifiers	User-Define Identifiers
int	printf	KMS_PER_MILE
void	scanf	miles
float		kms
double		sum
return		sum

- **NOTE: Sum, sum, SUM are viewed by the compiler as different identifiers**

Placeholders in Format String

Placeholder	Variable Type	Function Use
%c	char	printf / scanf
%d	int	printf / scanf
%f	float	printf / scanf
%f	double	printf
%lf	double	scanf

From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Placeholders in Format String

```
#include<stdio.h>
int main(){
 int sum = 2;
 float a = 3.2, r = 5.2;
 printf (“The area is %f”, a);
 scanf(“ %f ”,&r);
 printf (“the result is %d”, sum);
 printf (“the number is %f”, num);
 return 0;
}
```


From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Arithmetic Operations

Arithmetic Operator	Meaning	Examples
+	addition	5 + 2 is 7
-	subtraction	5 - 2 is 3
*	multiplication	5 * 2 is 10
/	division	5 / 2 is 2
%	Remainder or Mod	5 % 2 is 1

Arithmetic Operations

Results of / and % operations

2 / 15 = 0	int / int = int
16 / 3 = 5	12/3= 4 , 9/8=1
4 / 0 undefined	int/float =float , float/int=float
2 % 5 = 2	float/float=float
5 % 4 = 1	9/8.0=1.125000
15 % 0 undefined	9.0/8=1.125000
	9.0/8.0=1.125000

Arithmetic Operations - Example

```
#include<stdio.h>
int main(){
 double k, m;
 k = 9/6;
 m = 9/6.0;
 printf("k=%f \nm= %f", k,m);
 return 0;
}
```

Output:

k=1.000000

m=1.500000

From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

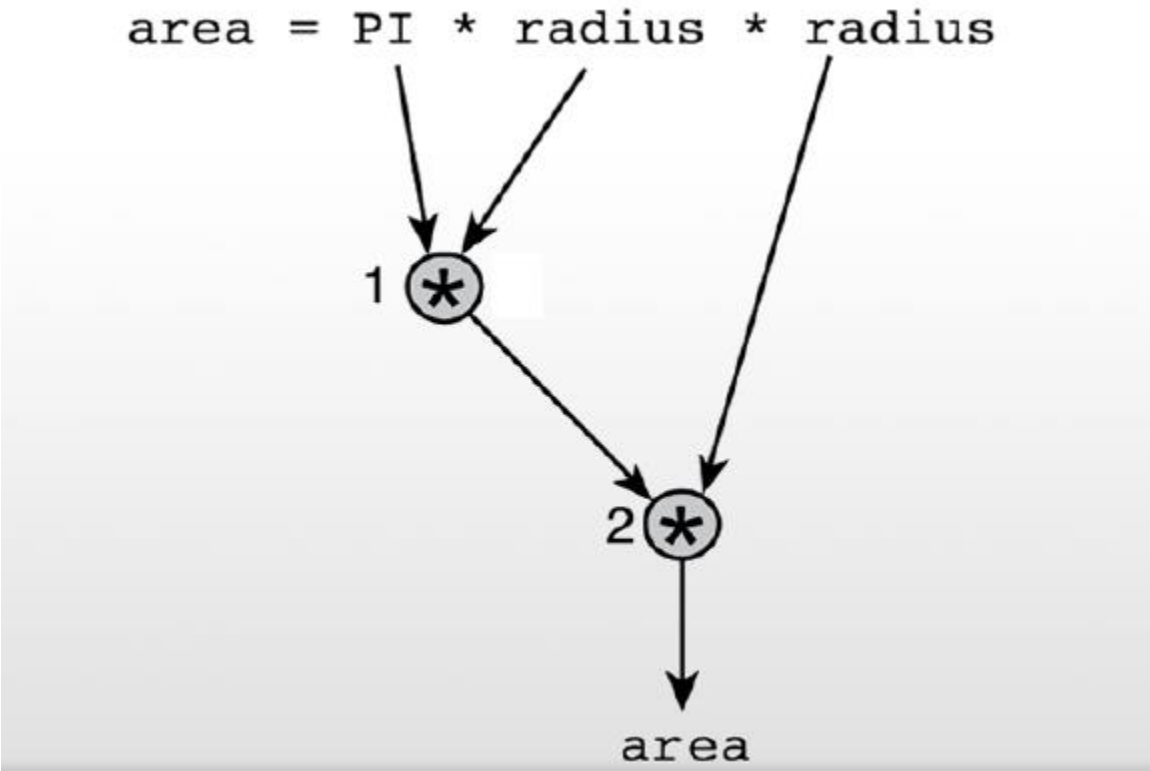
Arithmetic Operations

- Precedence Rules:
  - ()
  - * / %
  - + -

From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Arithmetic Expressions

- Example 1 :
  Evaluate `area = PI * radius * radius;`



From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Arithmetic Expressions

- Example 1 :
  Evaluate `area = PI * radius * radius;`
- Let `PI= 3.14159 , radius=2.0`

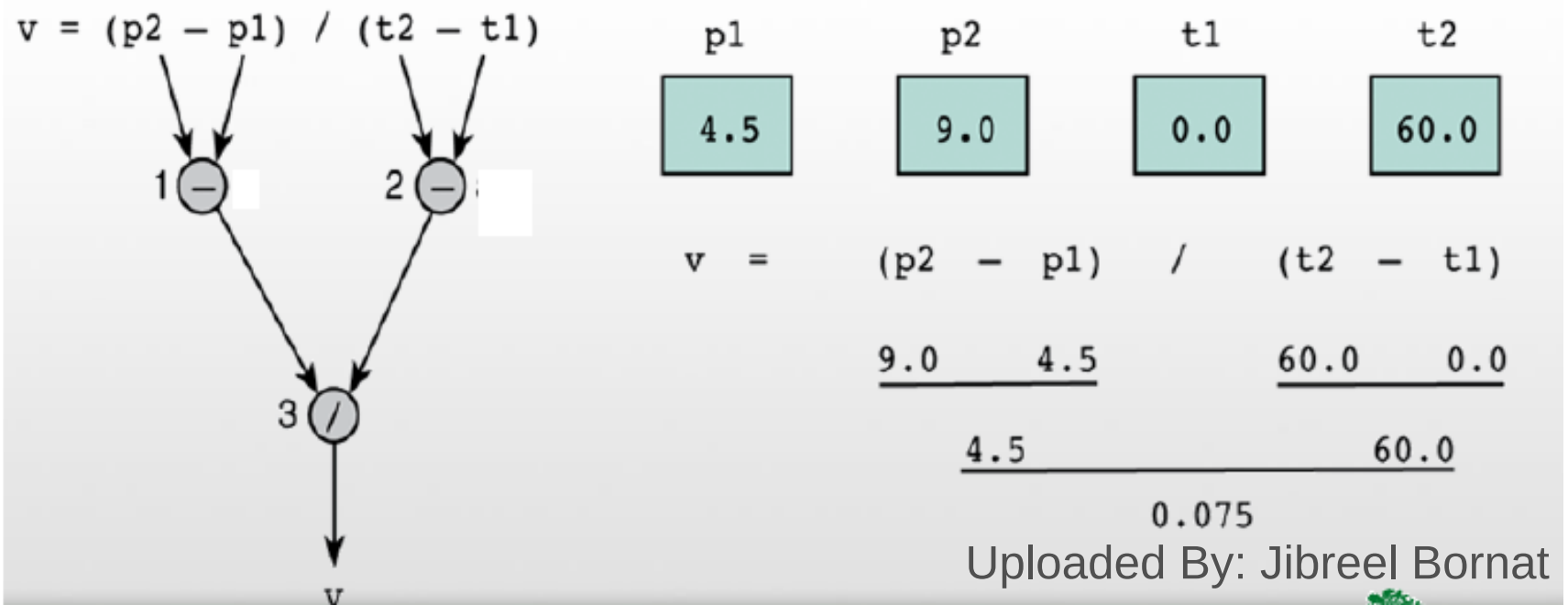
```
area    =     PI     *   radius    *   radius
            3.14159        2.0          2.0
                 6.28318
                                  12.56636
```

Arithmetic Expressions

- Example 2:

Evaluate $v = \dfrac{p2-p1}{t2-t1}$

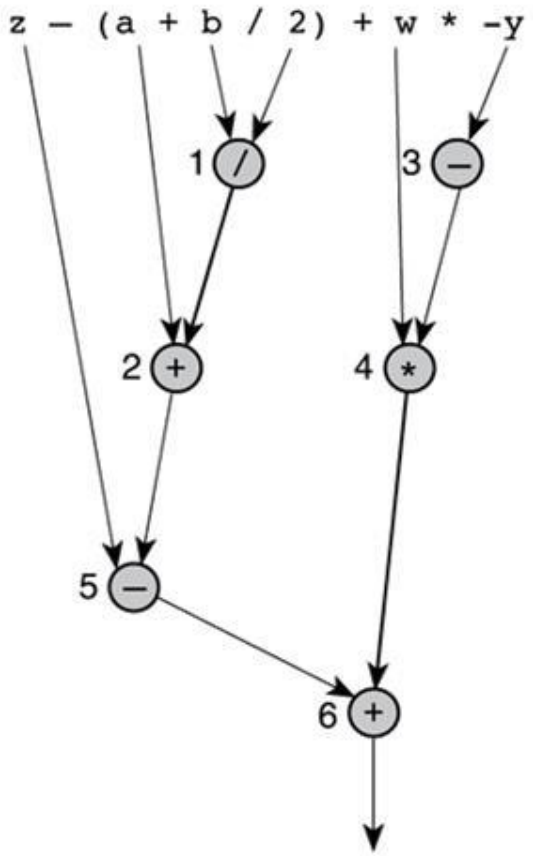
let P1=4.5 ,P2=9.0, t1=0.0, t2=60.0

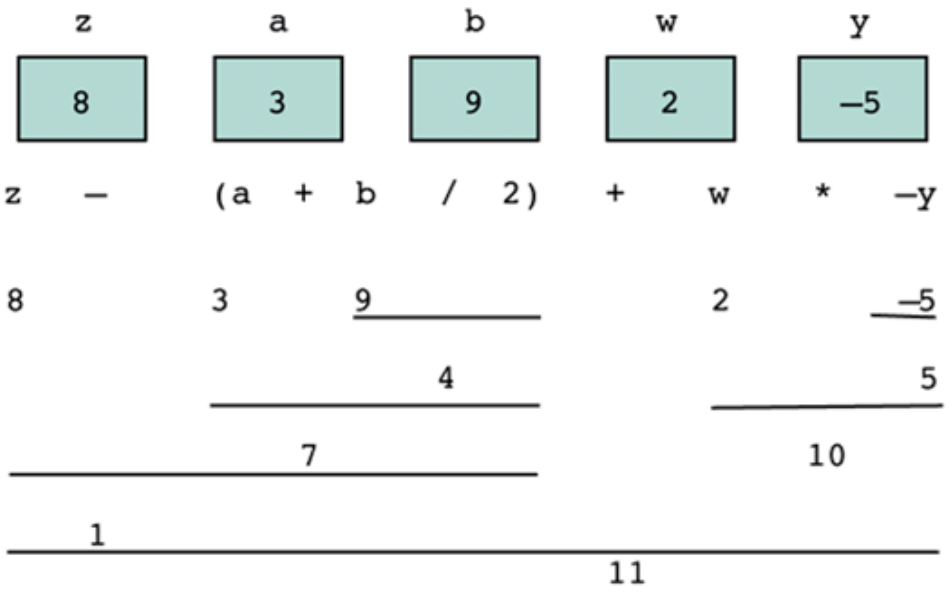


From the material of Mr. Abdallah Karakra, 2016, I

Arithmetic Expressions

- Example 3: Evaluate: `z -(a + b / 2) + w * -y;`





From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Arithmetic Expressions

Mathematical Formula as C Expression

Mathematical Formula	C Expression
b^2-4ac	b * b - 4 * a * c
a + b - c	a + b - c
$\frac{a+b}{c+d}$	(a + b) / (c + d)
$\frac{1}{1+x^2}$	1 / (1 + x * x)
a x -(b + c)	a * -(b + c)

From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Example

- **Write a complete C program that prompts the user to enter the radius of a circle and displays the circumference. Circumference=2πr**

```
#include <stdio.h>
#define PI 3.14159
int()
{
  float radius, circum;
  printf("Please enter radius of circle> ");
  scanf("%lf", &radius);
  circum = 2 * PI * radius;
  printf("The circumference is %.2f.\n", circum);
  return  0;
}
```


From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Casting – Type Conversion

- Type cast: converting a variable or expression from one data type to another during the application run.
- Example:
- `float f = 3/2;`
  - This will store 1.0 in f because 3 and 2 are considered integers.
- Casting:
  - `float f = (float)3/2;`

From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Casting – Type Conversion

- `int x = 97;`
- `printf("%c", (char)x);`

- `float y = 5.5;`
- `printf("%d", (int)y);`

- `float n = (int)(9 * 0.5);`

From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Formatting Output

- `int x= 4678,  y=3 ,  z=19;`

- `printf ("%d%d%d", x,y,z);`
- Output: 4678319

- `printf ("%d %d %d", x,y,z);`
- Output: 4678 3 19

Formatting Output

- `int x= 4678,  y=3 ,  z=19;`

- `printf ("%7d%5d%6d", x,y,z);`
- Output: ▒▒▒4678▒▒▒▒▒3▒▒▒▒▒19

Formatting Output

- `float x=56.2757, y=2.3849, z=114.2;`
- `printf ("%8.3f%-7.2f%7.4f",x,y,z);`
- **Output:** ▒▒56.276 2.38▒▒▒114.2000
- `double a= 38.56, b= 201.117;`
- `printf("Is it%6.1f%9.4f", a, b);`
- **Output:** `Is it` ▒▒38.6▒201.1170

From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Formatting Output

- `float x=333.256;`
- `printf("%0.2f",x);`
- Output: `333.26`

Formatting Output

Value	Format	Displayed Output		Value	Format	Displayed Output
234	%4d	▒234		-234	%4d	-234
234	%5d	▒▒234		-234	%5d	▒-234
234	%6d	▒▒▒234		-234	%6d	▒▒-234
234	%1d	234		-234	%2d	-234

From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Formatting Output

Value	Format	Displayed Output	Value	Format	Displayed Output
3.14159	%5.2f	▒3.14	3.14159	%4.2f	3.14
3.14159	%3.2f	3.14	3.14159	%5.1f	▒▒3.1
3.14159	%5.3f	3.142	3.14159	%8.5f	▒3.14159
.1234	%4.2f	0.12	-.006	%4.2f	-0.01
-.006	%8.3f	▒▒-0.006	-.006	%8.5f	-0.00600
-.006	%.3f	-0.006	-3.14159	%.4f	-3.1416

From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Example

- Write a program to reverse any two digits number?

```
#include <stdio.h>
int main()
{
    int num;
    int rem;
    int rev;
    int tens;
    printf("Please enter two digits number");
    scanf ("%d",&num);
    tens= num / 10;
    rem=num % 10;
    rev= rem * 10;
    rev= rev+ tens;
    printf ("the result is %d", rev);
    return 0;
}
```

Common programming errors

- Syntax Errors
  - is a mistake in the syntax.
- E.g.,
  - missing semicolon
  - undeclared variable
  - last comment is not closed because of blank in */ close-comment sequence

From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Common programming errors

- Logic Errors
- an error caused by following an incorrect algorithm.
- E.g.,:
  - sum = x-y (minus instead of plus)

Common programming errors

- Run-time Errors
  - an attempt to perform an invalid operation, detected during program execution.
- E.g.,:
  - result = x / 0 (undefined)

Files

- Declare a file pointer variable
  - `FILE *ftp_in ;` /* pointer to input file */
  - `FILE *ftp_out;` /* pointer to output file */
- The calls to open the files (fopen)
  - `ftp_in = fopen("distance.dat", "r" ) ;`
  - `ftp_out = fopen("distance.out", "w") ;`
- Use of functions
  - `fscanf(ftp_in, "%lf", &miles);`
  - `fprintf(ftp_out, "The distance in miles is %.2f. \n", miles);`
- End of use
  - `fclose(ftp_in);`
  - `fclose(ftp_out);`

From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Files

- Example: Write a program that reads two integers from a file called input.txt, then find their sum and save the result into another file called output.txt.

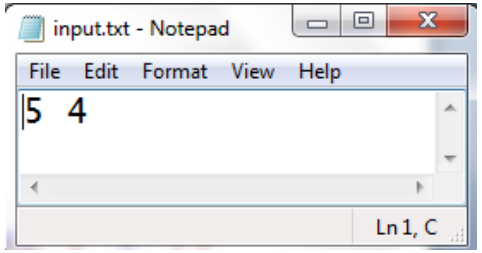
Files

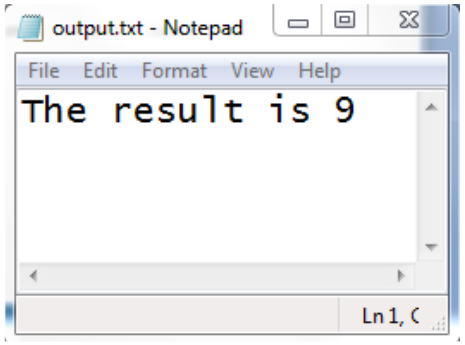
```c
#include <stdio.h>
int main()
{
    FILE *fpt_in, *fpt_out;
    int num1,num2;
    int sum;

    fpt_in = fopen ("input.txt","r");
    fpt_out = fopen ("output.txt","w");

    fscanf (fpt_in,"%d%d",&num1,&num2);
    sum=num1+num2;

    fprintf(fpt_out,"The result is %d",sum);
    fclose(fpt_in);
    fclose(fpt_out);
    return 0;
}
```





From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Files

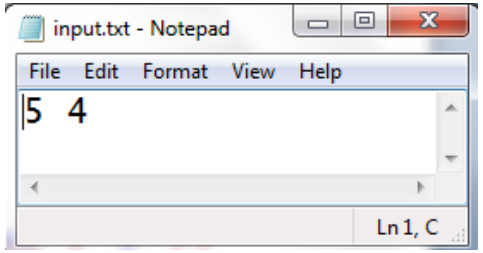
- Example: Write a program that reads two integers from a file called input.txt, then find their sum and print it to the screen.

```
int main()
{
    FILE *fpt_in;
    int num1,num2;
    int sum;

    fpt_in = fopen ("input.txt","r");

    fscanf (fpt_in,"%d%d",&num1,&num2);
    sum=num1+num2;

    printf("The result is %d",sum);
    fclose(fpt_in);
    return 0;
}
```




```
The result is 9
Process returned 0 (0x0)   execution time : 0.009 s
Press any key to continue.
```


From the material of Mr. Abdallah Karakra, 2016, Birzeit University.

Escape sequences

- Escape Sequence causes the program to escape from the normal interpretation of a string, so that the next character is recognized as having a special meaning.
- The back slash "\" character is called the "**Escape Character**".
- The escape sequence includes the following:
- **\n => new line**
- **\t => tab**
- **\" => double quotations**
- **\\=> back slash**