

جاء

Fundamentals of Programming

Bu-Ali Sina University, Tuyserkan Faculty of Engineering



Mohammad R. Tayyebi

Undergraduate, BASU

Blogger, Sariab.ir



Introduction

Session 1, Dec 14 2019

Session Goals:

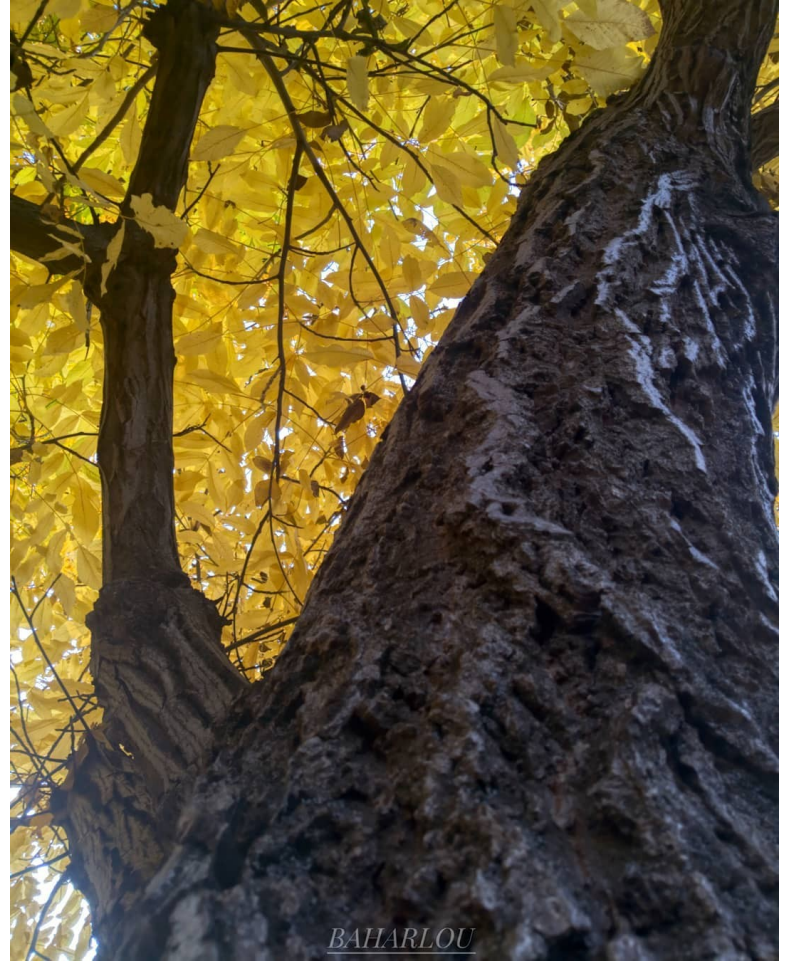
Greetings and Rules,

Algorithmic Thinking,

Enjoying Nature,

Become Familiar with Some Keywords

Greetings



Rules

2nd : Keep the bathroom trips to minimum

3rd : Don't be too comfortable [that you fall sleep and start snoring]

4th : No kids

Rules

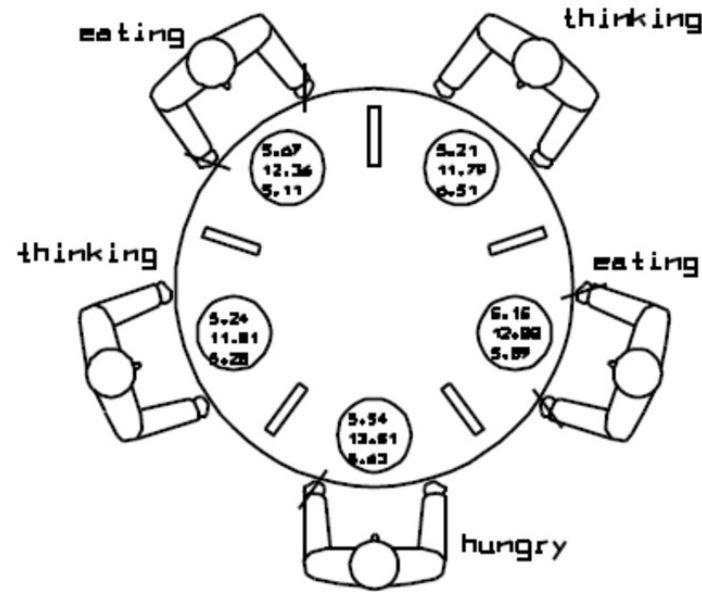
1st and the most important: ***ENJOY THE CLASS!***

5th : Listen carefully, then write it down

6th : Practice, Try, Try, Try...

7th : “If you are born *philosopher*, it’s not your fault. But if you die *philosopher*, it’s your mistake.” - William Henry Gates III; And notice that old programmers never die, they just cast into **void**!

Dining philosophers problem



This court is now in session!

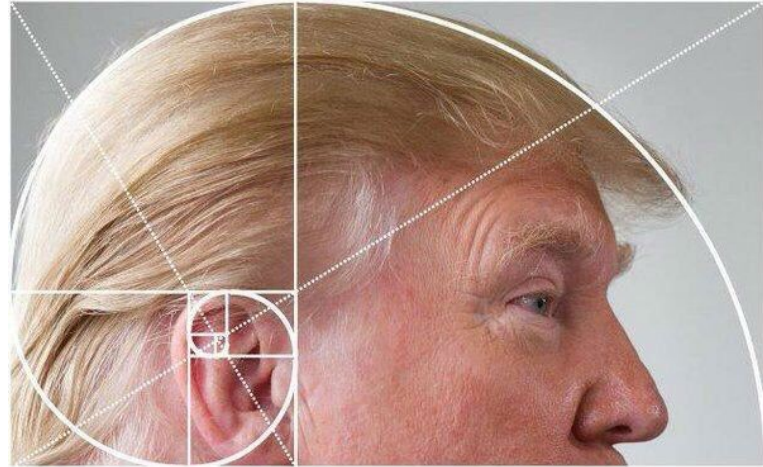
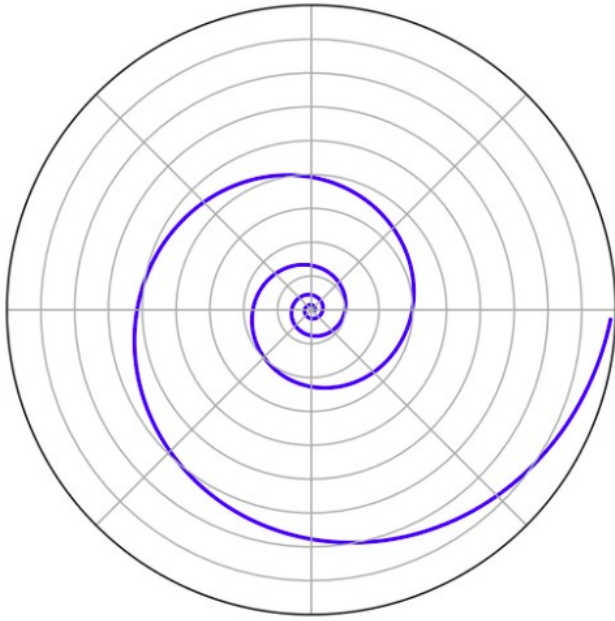


محَلّ

ارْحَمُوا عَزِيزاً ذَلَّ
وَ غَنِيّاً افْتَقَرَ
وَ عَالِماً ضَاعَ فِي زَمَانٍ جُهَالٍ.

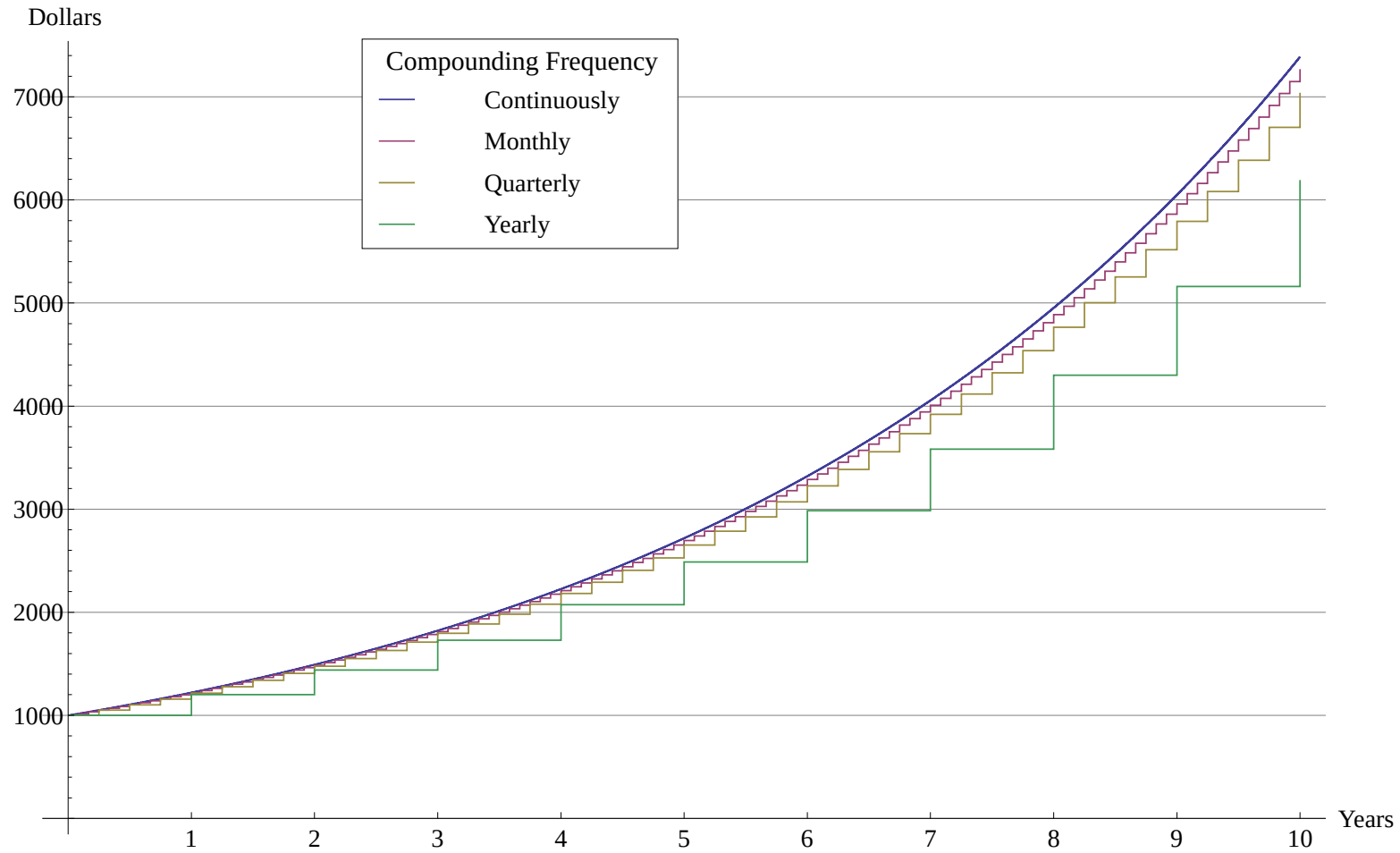
گفت پیغمبر که بر این سه گروه رحم آرید ار ز سنگید ار ز کوه/ آنکه او بعد از عزیزی خوار شد و آنکه بد با مال و بی دینار شد
و آن سوم آن عالمی کاندز جهان مبتلا گردد میان ابلهان/ ز آنکه از عزت به خواری آمدن همچو قطع عضو باشد از بدن

Logarithmic spiral



Compound interest

An account starts with \$1.00 and pays 100 percent interest per year. If the interest is credited once, at the end of the year, the value of the account at year-end will be \$2.00. If the interest is credited twice in the year, the interest rate for each 6 months will be 50%, so the initial \$1 is multiplied by 1.5 twice, yielding $\$1.00 \times (1.5^2) = \2.25 at the end of the year. Compounding quarterly yields $\$1.00 \times (1.25^4) = \$2.4414\dots$, and compounding monthly yields $\$1.00 \times (1 + 1/12)^{12} = \$2.613035\dots$ If there are n compounding intervals, the interest for each interval will be $100\%/n$ and the value at the end of the year will be $\$1.00 \times (1 + 1/n)^n$.

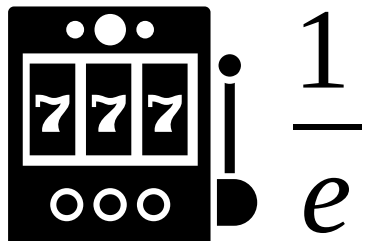


Euler's constant

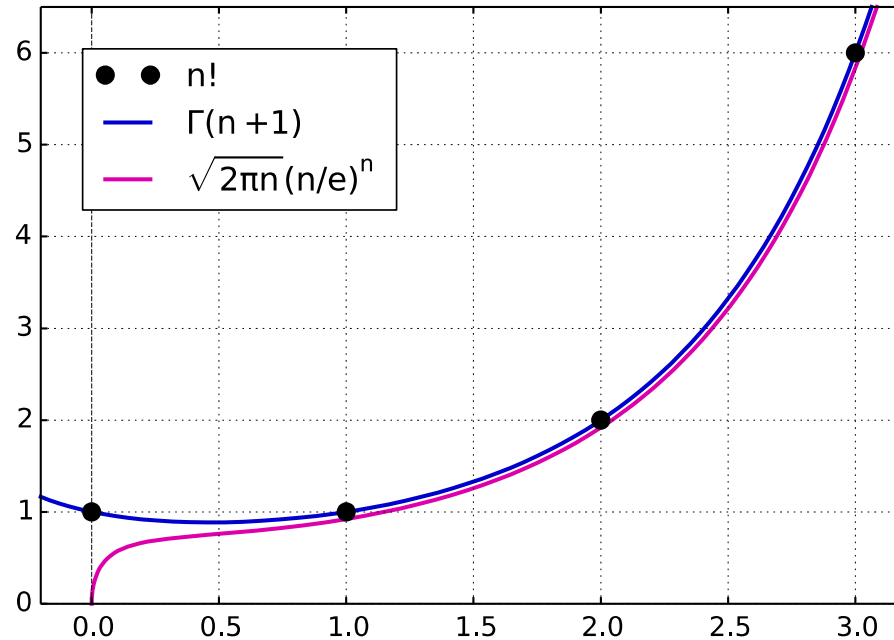
2.7 1828 1828 45 90 45

تاریخچه عدد طبیعی

اولین اشاره به این عدد، در جدولی که در ضمیمه مقاله مربوط به لگاریتم جان نپر در سال ۱۶۱۸ انتشار یافته بود مشاهده می‌شود. با این حال، این مقاله توضیحی راجع به این عدد نمی‌داد بلکه تنها لیستی از لگاریتم‌های حساب شده در مبنای این عدد را نشان می‌داد. به نظر می‌رسد که این جدول توسط ویلیام اوترد (مخترع خط‌کش محاسبه) تهیه شده است. اما «کشف» این عدد توسط ژاکوب برنولی به انجام رسید.



Stirling's Approximation



Euler's constant

$$\sum_{n=0}^{\infty} \frac{1}{n!}$$

$$\lim_{n \rightarrow \infty} \left(1 + \frac{1}{n}\right)^n$$

Code

```
#include <stdio.h>

long factorial(int i) {
    long result = 1;
    for (int j = 1; j <= i; ++j)
        result *= j;
    return result;
}
```

```
int main ()
{
    long double e = 0;
    int i;
    for (i = 0; i <= 32; i++) {
        e = ( 1.0 / factorial(i) ) + e;
    }
    printf("%.64Lf\n", e);
}
```

```
#include <stdio.h>

long factorial(int n) {
    long result = 1;
    for (int i = 1; i <= n; ++i)
        result *= i;
    return result;
}

int main ()
{
    long double e = 0;
    int i;

    for (i = 0; i <= 32; i++) {
        e = ( 1.0 / factorial(i) ) + e;
    }

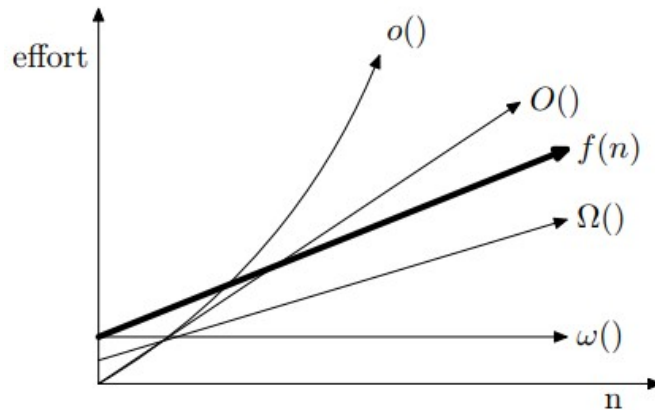
    printf("%.64Lf\n", e);
}
```

Algorithm Complexity (Big Ohhh!)

$O(N^2)$

$$f(n) = O(g(n)) \Leftrightarrow (\exists c)(\exists n_0)(\forall n > n_0)(f(n) \leq c \cdot g(n))$$

$$f(n) = \omega(g(n)) \Leftrightarrow (\forall c)(\exists n_0)(\forall n > n_0)(f(n) > c \cdot g(n))$$



$O(n)$: in worst situations, your algorithm has a complexity of n

$\Omega(n)$: in best case, your algorithm has a complexity of n

$\Theta(n)$: in every situation, your algorithm has a complexity of n

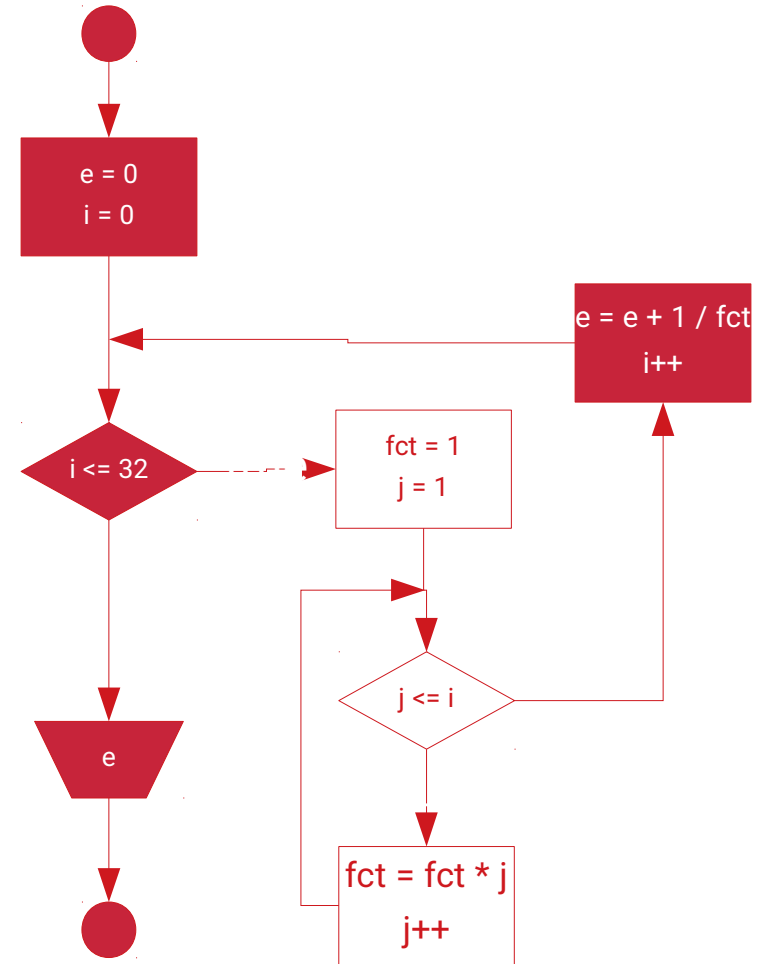
How to run the code

```
g++ calc0.cpp -o calc0.o ; time ./calc0.o
```

```
Terminal
File Edit View Search Terminal Help
$ ~/Desktop/funds/euler g++ calc0.cpp -o calc0.o ; time ./calc0.o
2.7182818284590452211166994311852818100305739790201187133789062500

real    0m0.001s
user    0m0.001s
sys     0m0.000s
```

Flowchart



Optimized Code

```
long double e = 0, f = 1;
int i;
for (i = 8; i >= 1; i--) {
    f = f * i;
    e = e + f;
}
e = e / f;
```

```
#include <stdio.h>
#include <iomanip>

int main ()
{
    long double e = 0, f = 1;
    int i;
    for (i = 28; i > 0; i--) {
        f *= i;
        e += f;
    }
    e /= f;
    printf("%.64Lf\n", e);
}
```

Optimized Code Traced

i	f	e	<u>f = f * i</u>	<u>e = e + f</u>	<i>e / f</i>
8	1	0	1 * 8 = 8	0 + 8 = 8	1.00000
7	8	8	8 * 7 = 56	8 + 56 = 64	1.14286
6	56	64	56 * 6 = 336	64 + 336 = 400	1.19048
5	336	400	1680	2080	1.23810
4	1680	2080	6720	8800	1.30952
3	6720	8800			1.43651
2	20160	28960			1.71825
1	40320	69280			2.71825
0			END OF LOOP	END OF LOOP	e = e / f = 2.7182539682

Mathematical Method

```
12  def main(n):
13      e = 0.0
14      f = 1.0
15      r = -1
16      while r <= n:
17          e += P (n, r)
18          r += 1
19      e /= P (n, r - 2)
20      return e
```

<https://gist.github.com/tayyebi/5bd1bfedbc71acb5801c049c5054de63>

Programming Languages

A programming language is a formal language, which comprises a set of instructions that produce various kinds of output. Programming languages are used in computer programming to implement algorithms. Most programming languages consist of instructions for computers. -

[Wikipedia](#)

C++

```
#include <stdio.h>
#include <iomanip>
```

```
int main ()
{
    long double e = 0, f = 1;
    int i = 28;
    while (i>0) {
        f *= i;
        e += f;
        i--;
    }
    e /= f;
    printf("%.64Lf\n", e);
}
```

calc1.cpp

x

```
1  #include <stdio.h>
2  #include <iomanip>
3
4  int main ()
5  {
6      long double e = 0, f = 1;
7      int i = 28;
8      while (i>0) {
9          f *= i;
10         e += f;
11         i--;
12     }
13     e /= f;
14     printf("%.64Lf\n", e);
15 }
```

Python

```
def main():  
    e = 0.0  
    f = 1.0  
    i = 28  
    while i > 0:  
        f *= i  
        e += f  
        i = i - 1  
    e /= f  
    return e  
  
if __name__ == "__main__":  
    print (main())
```

```
calc1.py x  
1  def main():  
2      e = 0.0  
3      f = 1.0  
4      i = 28  
5      while i > 0:  
6          f *= i  
7          e += f  
8          i = i - 1  
9      e /= f  
10     return e  
11  
12  if __name__ == "__main__":  
13     print (main())
```

Java

```
public class my_main{  
    public static void main(String[] args){  
        double e = 0, f = 1;  
        int i = 28;  
        while (i>0) {  
            f *= i;  
            e += f;  
            i--;  
        }  
        e /= f;  
        System.out.println(e);  
    }  
}
```

```
calc1.java x  
1  public class my_main{  
2      public static void main(String[] args){  
3          double e = 0, f = 1;  
4          int i = 28;  
5          while (i>0) {  
6              f *= i;  
7              e += f;  
8              i--;  
9          }  
10         e /= f;  
11         System.out.println(e);  
12     }  
13 }
```

JavaScript

```
var e = 0.0, f = 1.0, i = 28;
```

```
while (i>0) {  
    f *= i;  
    e += f;  
    i--;  
}
```

```
e /= f;
```

```
console.log(e);
```

```
calc1.js x  
1  var e = 0.0, f = 1.0, i = 28;  
2  
3  while (i>0) {  
4      f *= i;  
5      e += f;  
6      i--;  
7  }  
8  
9  e /= f;  
10  
11 console.log(e);
```

PHP

```
<?php
$e = 0;
$f = 1;
$i = 28;
while ($i > 0) {
    $f *= $i;
    $e += $f;
    $i--;
}
$e /= $f;
print($e);
?>
```

calc1.php

x

```
1  <?php
2  $e = 0;
3  $f = 1;
4  $i = 28;
5  while ($i > 0) {
6      $f *= $i;
7      $e += $f;
8      $i--;
9  }
10 $e /= $f;
11 print($e);
12 ?>
```

Go

```
package main
```

```
import "fmt"
```

```
func main() {
```

```
    var e = 0.0
```

```
    f := 1.0
```

```
    var i float64
```

```
    i = 28
```

```
    for i > 0 {
```

```
        f *= i
```

```
        e += f
```

```
        i--
```

```
    }
```

```
    e /= f
```

```
    fmt.Println(e)
```

```
}
```

calc1.go

x

```
1 package main
```

```
2
```

```
3 import "fmt"
```

```
4
```

```
5 func main() {
```

```
6
```

```
7     var e = 0.0
```

```
8     f := 1.0
```

```
9     var i float64
```

```
10    i = 28
```

```
11
```

```
12    for i > 0 {
```

```
13        f *= i
```

```
14        e += f
```

```
15        i--
```

```
16    }
```

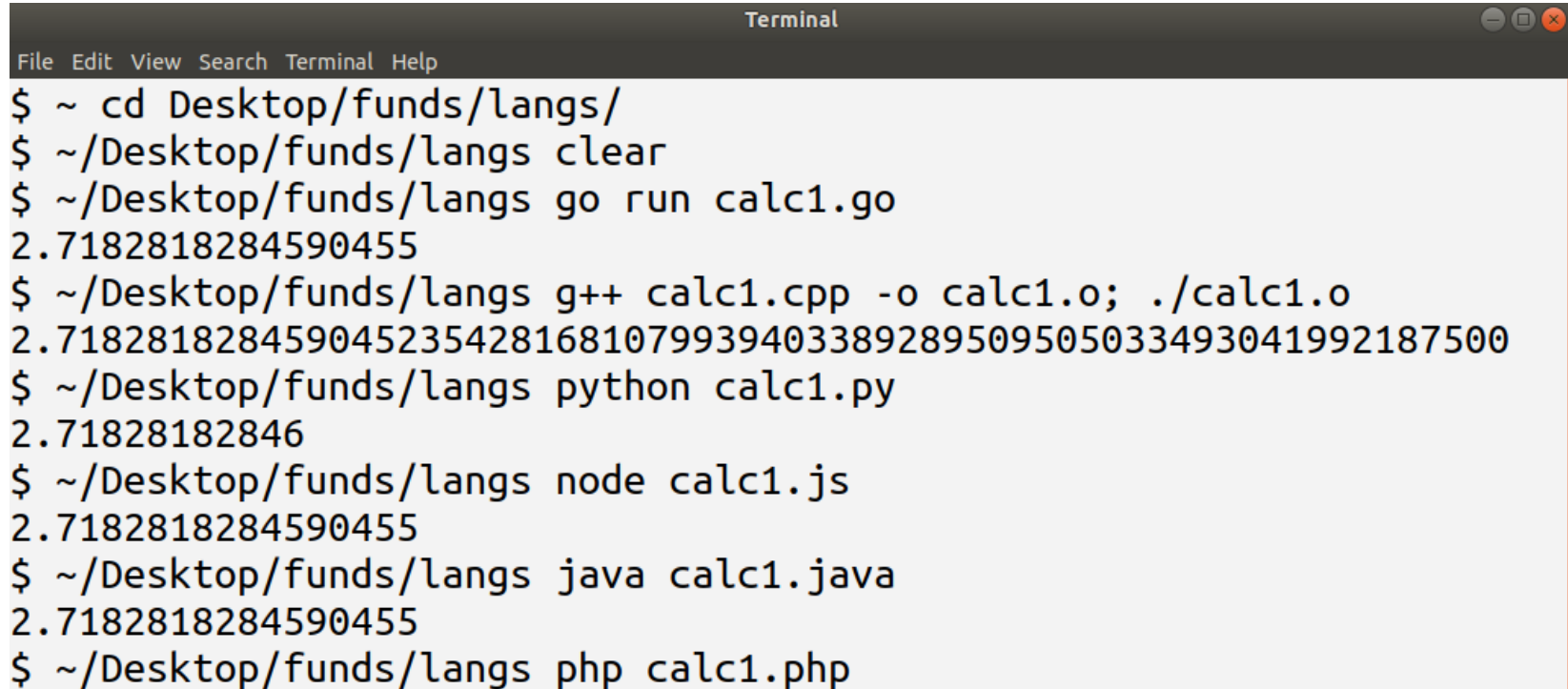
```
17    e /= f
```

```
18    fmt.Println(e)
```

```
19
```

```
20 }
```

Linux is your friend



A terminal window titled "Terminal" with a menu bar (File, Edit, View, Search, Terminal, Help) and standard window controls. It displays a series of commands and their outputs for different programming languages.

```
File Edit View Search Terminal Help
$ ~ cd Desktop/funds/langs/
$ ~/Desktop/funds/langs clear
$ ~/Desktop/funds/langs go run calc1.go
2.7182818284590455
$ ~/Desktop/funds/langs g++ calc1.cpp -o calc1.o; ./calc1.o
2.7182818284590452354281681079939403389289509505033493041992187500
$ ~/Desktop/funds/langs python calc1.py
2.71828182846
$ ~/Desktop/funds/langs node calc1.js
2.7182818284590455
$ ~/Desktop/funds/langs java calc1.java
2.7182818284590455
$ ~/Desktop/funds/langs php calc1.php
```

Whats Next

Tomorrow

CPP: Main syntax, Functions, Variables, and Arrays

Sepas <3

Contacts:

<http://kouy.ir/tayyebi>

Proudly a Sariab Blogger:

www.sariab.ir, info@sariab.ir



Reference and Images:

[https://en.wikipedia.org/wiki/E_\(mathematical_constant\)](https://en.wikipedia.org/wiki/E_(mathematical_constant))

<https://www.onstageblog.com/onscreen/2018/6/13/9-rules-for-the-movie-theater>

<https://www.sne-journal.org/benchmarks/c10>

<https://www.commpro.biz/court-is-in-session-the-law-of-marketing-small-brands-cannot-overlook/36419114-hand-about-to-bang-gavel-on-sounding-block-in-the-court-room/>

<https://codereview.stackexchange.com/questions/33015/>

<https://stackoverflow.com/questions/27873104/>

Legend of Dennis Ritchie

Session 2, Dec 15 2019

Session Goals:

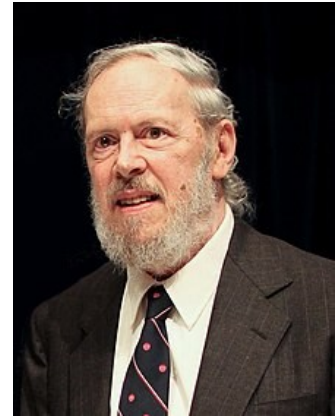
Knowing of C++ Fundamentals,

Knowing of C++ Main Syntax,

Knowing of Data Types and Variables,

Knowing of Arrays, Strings, and Multidimensional Implementation

Knowing of Pointers



Session 1 - Review

- Even these days we have machine resources limitations
- Machine is the computer
- Formal language means being human friendly
- “Teachers who make math (physics) boring are criminals” -
Walter Lewin, MIT
- Do we need flowcharts?
- WTH is Linux?

Variables

char -127 to 127

unsigned char 0 to 255

int -32,767 to 32,767

double 64b Ten digits of precision

long double 80b Ten digits of precision

Define Variables

```
int i,j,l;
```

```
short int si;
```

```
unsigned int ui;
```

```
double balance, profit, loss;
```

Local Variables

```
void func1(void)
```

```
{
```

```
    int x;
```

```
    x = 10;
```

```
}
```

```
void func2(void)
```

```
{
```

```
    int x;
```

```
    x = -199;
```

```
}
```

Example 1

```
#include <iostream>
#include <string.h>
#include "stdio.h"
using namespace std;

void f(void)
{
    int t;
    scanf("%d%c", &t);
    if(t==1) {
        char s[80];
        printf("Enter name:");
        cin >> s;
        printf("%s",s);
    }
}

int main()
{
    f();
}
```


Define Functions

```
void f(void);
```

```
int main(void)
```

```
{
```

```
    f();
```

```
}
```

```
void f(void)
```

```
{
```

```
    // Something
```

```
}
```

Global Variables

```
int count; /* count is global */
```

```
void func1(void);
```

```
void func2(void);
```

```
void func1(void)
```

```
{
```

```
    // Do something with var count
```

```
}
```

Example 2

```
#include <stdio.h>

int count; /* count is global */
void func1(void);
void func2(void);

int main(void)
{
    count = 100;
    func1();
    return 0;
}

void func1(void)
{
    int temp;
    temp = count;
    func2();
    printf("count is %d", count); /* will print 100 */
}

void func2(void)
{
    int count;
    for(count=1; count<10; count++)
        putchar('.');
}
```

Selection Statements - If

```
if (expression)
```

```
    statement;
```

```
else
```

```
    statement;
```

Nested Ifs

```
if(i)
{
    if(j) statement 1;
    if(k) statement 2;
    else statement 3;
}
else statement 4;
```

if-else-if Ladder

```
if (expression)
    statement;
else if (expression)
    statement;
else if (expression)
    statement;
.
.
.
else
    statement;
```

ternary operator of “?”

Exp1 ? Exp2 : Exp3

Single dimension arrays

```
type var_name[size];
```


Example 3

```
#include <stdio.h>
int main(void)
{
    int x[100]; /* this declares a 100-integer array */
    int t;
    /* load x with values 0 through 99 */
    for(t=0; t<100; ++t)
        x[t] = t;
    /* display contents of x */
    for(t=0; t<100; ++t)
        printf("%d ", x[t]);
    return 0;
}
```

Generating a Pointer to an Array

```
int *p;
```

```
int sample[10];
```

```
p = sample;
```

Null-Terminated Strings

`strcpy(s1, s2)` Copies s2 into s1.

`strcat(s1, s2)` Concatenates s2 onto the end of s1.

`strlen(s1)` Returns the length of s1.

`strcmp(s1, s2)` Returns 0 if s1 and s2 are the same; less than 0 if s1<s2; greater than 0 if s1>s2.

`strchr(s1, ch)` Returns a pointer to the first occurrence of ch in s1.

`strstr(s1, s2)` Returns a pointer to the first occurrence of s2 in s1.

Null-Terminated Strings

```
char e3[] = "Cannot open file\n";
```

Example 4

```
#include <stdio.h>
#include <string.h>
#include <iostream>

using namespace std;

int main(void)
{
    char s1[80], s2[80];
    cin >> s1; // gets(s1);
    cin >> s2; // gets(s2);
    cout << "lengths: " << strlen(s1) << " " << strlen(s2) << "\n";
    if(!strcmp(s1, s2)) printf("The strings are equal\n");
    strcat(s1, s2);
    printf("%s\n", s1);
    strcpy(s1, "This is a test.\n");
    cout << s1;
    if(strchr("hello", 'e')) printf("e is in hello\n");
    if(strstr("hi there", "hi")) printf("found hi");
    return 0;
}
```

Multidimensional Arrays

```
type name [Size1] [Size2] [Size3] . . . [SizeN];
```

Example 5

```
/* A simple Tic Tac Toe game. */  
#include <stdio.h>  
#include <stdlib.h>  
  
char matrix[3][3];  
/* the tic tac toe matrix */  
char check(void);  
void init_matrix(void);  
void get_player_move(void);  
void get_computer_move(void);  
void disp_matrix(void);
```

Example 5

```
int main(void)
{
    char done;
    printf("This is the game of Tic Tac Toe.\n");
    printf("You will be playing against the computer.\n");
    done = ' ';
    init_matrix();
    do{
        disp_matrix();
        get_player_move();
        done = check(); /* see if winner */
        if(done!= ' ') break; /* winner!*/
        get_computer_move();
        done = check(); /* see if winner */
    } while(done== ' ');
    if(done=='X') printf("You won!\n");
    else printf("I won!!!!\n");
    disp_matrix(); /* show final positions */
    return 0;
}
```


Example 5

```
/* Initialize the matrix. */
void init_matrix(void)
{
    int i, j;
    for(i=0; i<3; i++)
        for(j=0; j<3; j++) matrix[i][j] = ' ';
}
/* Get a player's move. */
void get_player_move(void)
{
    int x, y;
    printf("Enter X,Y coordinates for your move: ");
    scanf("%d%c%d", &x, &y);
    x--; y--;
    if(matrix[x][y] != ' '){
        printf("Invalid move, try again.\n");
        get_player_move();
    }
    else matrix[x][y] = 'X';
}
```

Example 5

```
/* Get a move from the computer. */
void get_computer_move(void)
{
    int i, j;
    for(i=0; i<3; i++)
    {
        for(j=0; j<3; j++)
            if(matrix[i][j]==' ') break;
        if(matrix[i][j]==' ') break;
    }
    if(i*j==9) {
        printf("draw\n");
        exit(0);
    }
    else
        matrix[i][j] = 'O';
}
```

Example 5

```
/* Display the matrix on the screen. */
void disp_matrix(void)
{
    int t;
    for(t=0; t<3; t++) {
        printf(" %c | %c | %c ",matrix[t][0],
            matrix[t][1], matrix [t][2]);
        if(t!=2) printf("\n---|---|---\n");
    }
    printf("\n");
}
/* See if there is a winner. */
```

Example 5

```
char check(void)
{
    int i;
    for(i=0; i<3; i++) /* check rows */
        if(matrix[i][0]==matrix[i][1] &&
           matrix[i][0]==matrix[i][2]) return matrix[i][0];
    for(i=0; i<3; i++) /* check columns */
        if(matrix[0][i]==matrix[1][i] &&
           matrix[0][i]==matrix[2][i]) return matrix[0][i];
    /* test diagonals */
    if(matrix[0][0]==matrix[1][1] &&
       matrix[1][1]==matrix[2][2])
        return matrix[0][0];
    if(matrix[0][2]==matrix[1][1] &&
       matrix[1][1]==matrix[2][0])
        return matrix[0][2];
    return ' ';
}
```

What Are Pointers?

A pointer is a variable that holds a memory address.

Pointer Variables

If a variable is going to hold a pointer, it must be declared as such.

```
type *name;
```

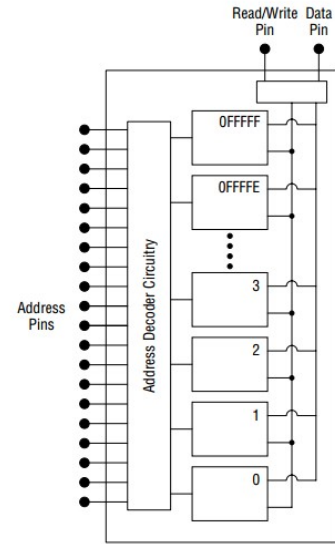


Figure 3-2: A RAM chip

The Pointer Operators

The `&` is a unary operator that returns the memory address of its operand.

```
m = &count;
```

The second pointer operator, `*`, is the complement of `&`. It is a unary operator that returns the value located at the address that follows.

```
q = *m;
```

Example 6

```
#include <iostream>
using namespace std;

int main(void)
{
    double x = 100.1, y;
    int *p;
    /* The next statement causes p (which is an
       integer pointer) to point to a double. */
    p = (int *)&x;
    /* The next statement does not operate as
       expected. */
    y = *p;

    cout << "y:  " << y << "\n"; /* won't output 100.1 */
    cout << "p:  " << p << "\n";
    cout << "x:  " << x << "\n";
    cout << "&x: " << &x << "\n";
    return 0;
}
```


Arrays of Pointers

The declaration for an int pointer array of size 10 is

```
int *x[10];
```

To assign the address of an integer variable called var to the third element of the pointer array, write

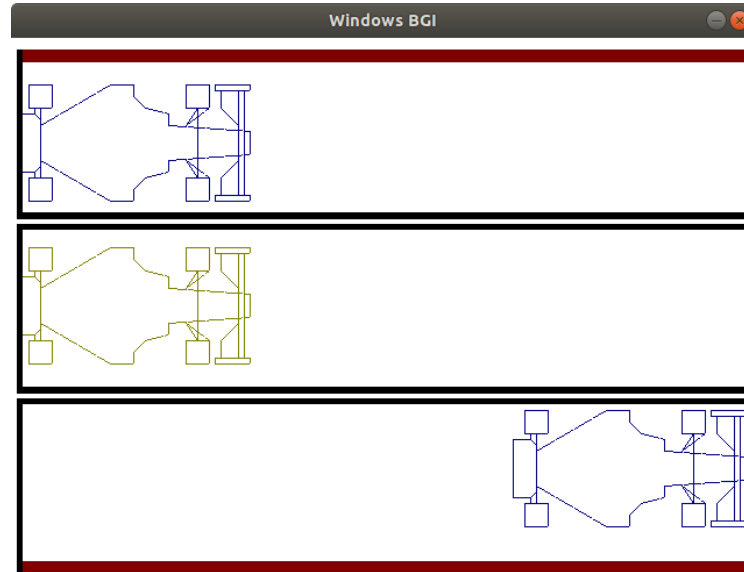
```
x[2] = &var;
```

To find the value of var, write

```
*x[2]
```

For more fun

<https://github.com/tayyebi/CarRacing>



Whats Next

Tomorrow

Object Oriented Design and Software Architecture

Sepas <3

Contacts:

<http://kouy.ir/tayyebi>

Sariab Website and Mailbox:

www.sariab.ir, info@sariab.ir

Sariab Instagram and Telegram:

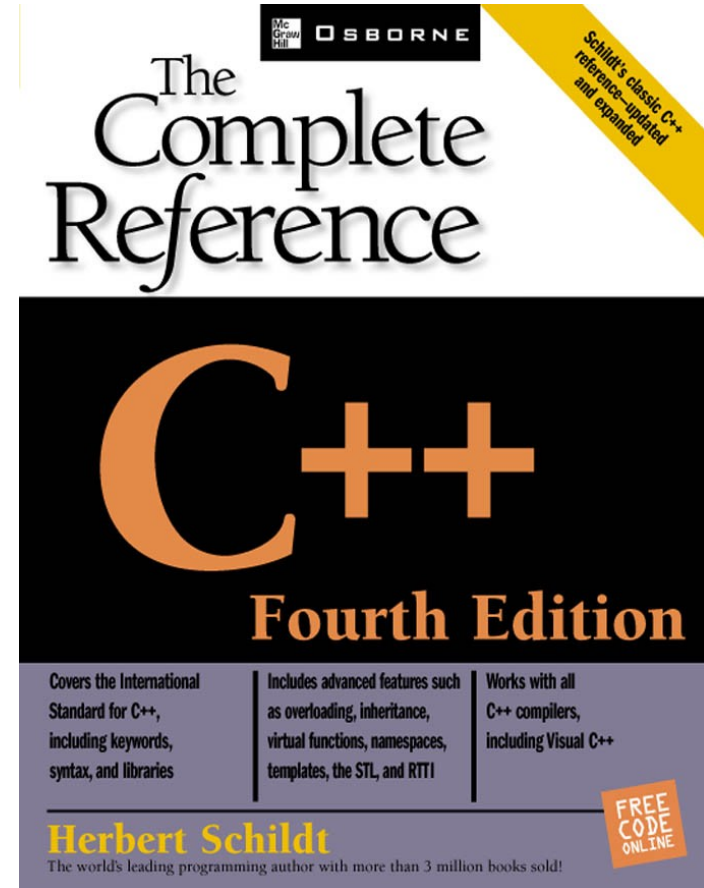
[@sariabbloggers](#)



Project's repository:

<https://github.com/tayyebi/fundamentals>

Reference:



Reference and Images:

<https://cs.stackexchange.com/questions/91579/how-is-memory-address-applied-to-address-pins-in-ram-chips/91588>