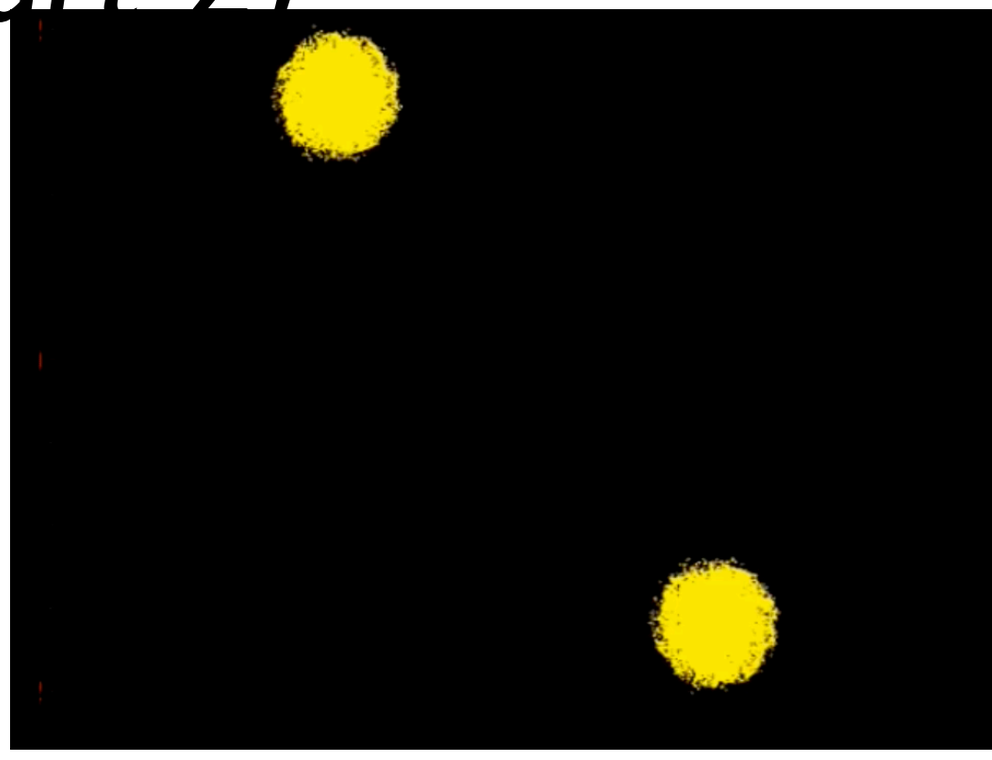
NagBody lectures: Data structures, algorithms and pointers (part 2)



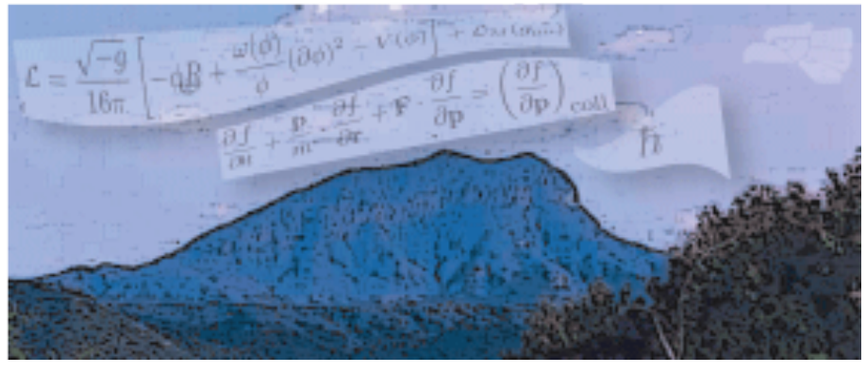
Mario Alberto Rodríguez-Meza

Instituto Nacional de Investigaciones Nucleares

Correo Electrónico: marioalberto.rodriguez@inin.gob.mx

http://bitbucket.org/rodriguezmeza

Seminario de investigación,
Departamento de Física,
Universidad de Guanajuato
3 de febrero al XX de junio de 2022
Sesiones virtuales (Zoom, Meet, etcétera)



References and material

- Cosmología numérica y estadística: NagBody kit (http://bitbucket.org/rodriguezmeza). Mario A. Rodríguez-Meza. And: https://github.com/rodriguezmeza/NagBody_lectures.git

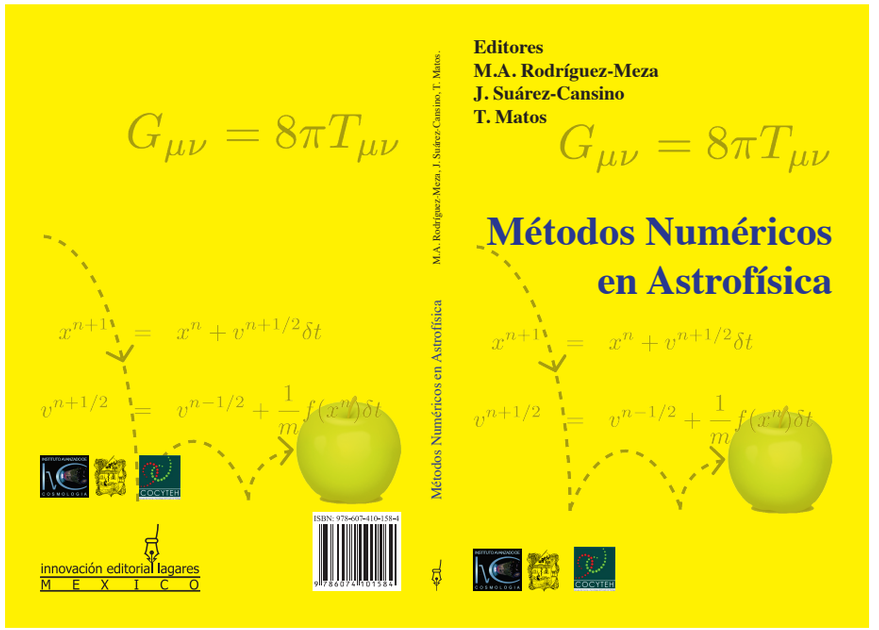
- Métodos numéricos en astrofísica, capítulo 1, Método de N-cuerpos en astrofísica. (https://www.researchgate.net/publication/316582859_Metodo_de_N-Cuerpos_en_Astrofisica)

- La estructura a gran escala del universo. Capítulo 22 en Travesuras cosmológicas de Einstein et al. https://www.researchgate.net/publication/316582400_La_estructura_a_gran_escala_del_universo_simulaciones_numericas

- https://www.researchgate.net/profile/Mario_Rodriguez-Meza

- https://www.researchgate.net/publication/314281416_Los_agujeros_negros_y_las_ondas_del_Dr_Einstein

- M.A. Rodriguez-Meza, Adv. Astron. 2012, 509682 (2012). arXiv: 1112.5201. (https://www.researchgate.net/publication/51967093_A_Scalar_Field_Dark_Matter_Model_and_Its_Role_in_the_Large-Scale_Structure_Formation_in_the_Universe)



Content:
Pointers. Part 2

- Data structures
- Algorithms
- Software engineering
- Pointer manipulation

Data structures

- A conceptual organization of information is the data structure concept.
- It is closely connected to algorithms. An algorithm is well defined procedures for solving problems.
- Efficiency of an algorithm depends on a well defined data structure where information must be formally organized.

Data structures

Data structures gives:

- Efficiency. Searching lists. Data organized as binary trees, linked lists, etc.
- Abstraction. Structures of data offers a level of abstraction. Stacks for example. What can we do with stacks. Restaurant commands. How to organize information to parallelize the algorithms.
- Reusability. Data structures are reusable because they tend to be modular and context free. In modules, their data structure is only modified or acted on by functions in the modules. We make a data structure of any type by using void pointers to data rather than maintaining private copies of the data in the data structure itself.
- A data structure together with basic operations is called an abstract datatype.
- Updates and maintenance is simpler.

Introduction to algorithms

- An algorithm is a well-defined procedures for solving problems.
- Do the job with the right amount of effort.

Introduction to algorithms

Algorithms (well-defined) gives:

- Efficiency. Certain types of problems occur often in computing, so there are already efficient ways to solve them. FFT, Blas, Quadpack, etc. Try to sort a number of entries in a an index for a book. Sorting is so common that…

- Abstraction. Provide a level of abstraction because many seemingly complicated problems cab be distilled into simpler ones for which well-known algorithms already exist. Shortest path to route a packet between two gateways in an internet.

- Reusability. Tridiagonal systems and spline. Distill the problem in simpler ones.

- A data structure together with basic operations is called an abstract datatype.

- Updates and maintenance is simpler.

Introduction to algorithms

Classification of algorithms (general) and approach to solve problems and design of algoritmhs:

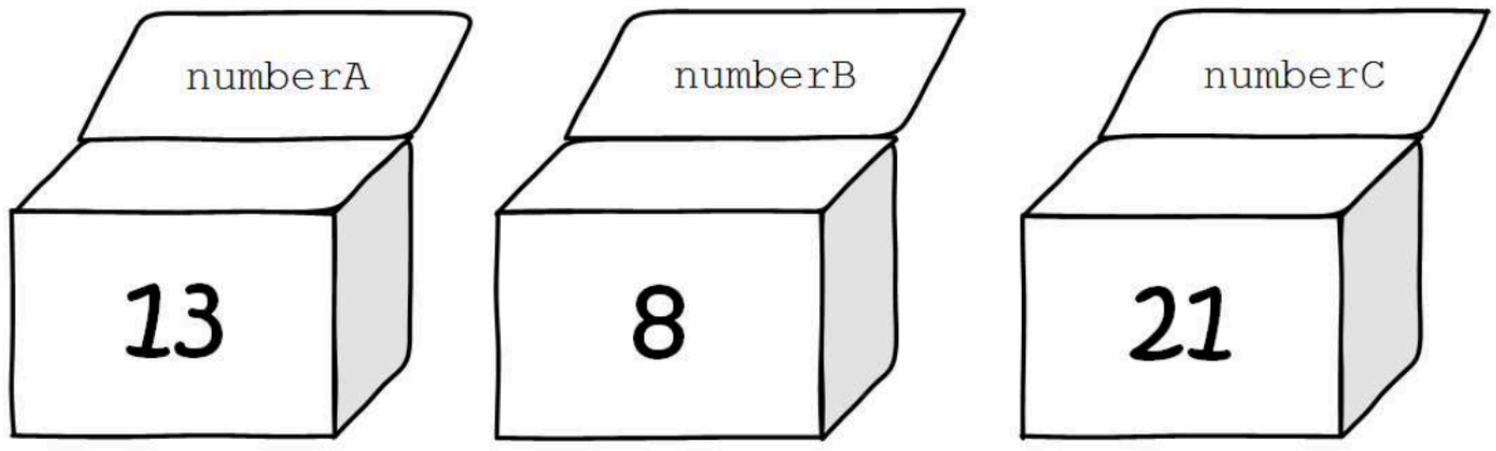
- Sorting. Randomize algorithms.
- Divide and conquer. Sorting also.
- Recursion.
- Parallelization.
- Approximation algorithms. Traveling sales man.

Software engineering

- A good understanding of data structures and algorithms is an important step to develop a well-crafted software.
- Do the job with the right amount of effort. Efficiency and efficacy.
- Modularity. Focus on the development of black boxes. Internals not intended to be seen by users of the modules. Users interact with the module only through a prescribed interface made public.
- The later is fundamental to data hiding and encapsulation. Abstract datatypes. User only can modified or operates on data structure using the functions defined for the datatype and made public in its header.
- Readability. See Barnes tree algorithms.
- Simplicity. See spline and splint of NR.
- Consistency. Coding conventions. Consistency gives readability and simplicity as well.

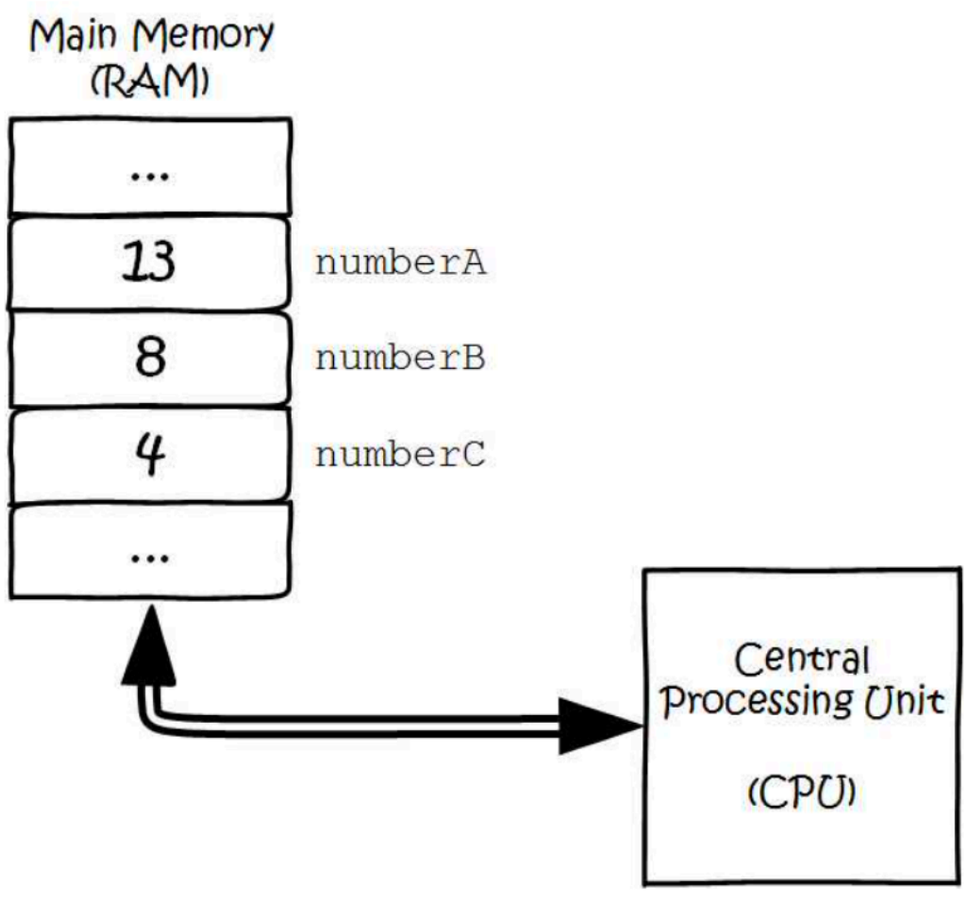
Variable concept again. Moving to pointers

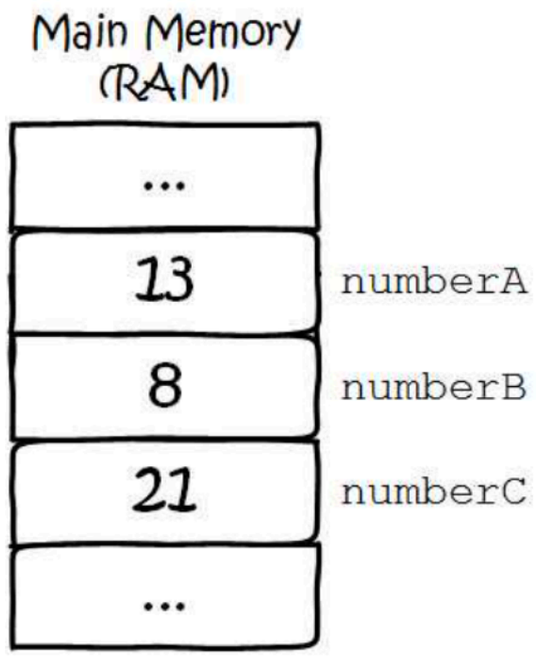
- A variable is a location in computer's main memory where a program can store its value and change it as program executes.



Variable concept again. Moving to pointers

- A variable is a location in computer's main memory where a program can store its value and change it as program executes.





Variable concept again. Moving to pointers

- A variable is one of the most important concept in programming.
- It helps to interact with the data stored in the main computer's memory.
- How many types there are:
  a. Integer.
  b. reals.
  c. Booleans.
  d. Characters

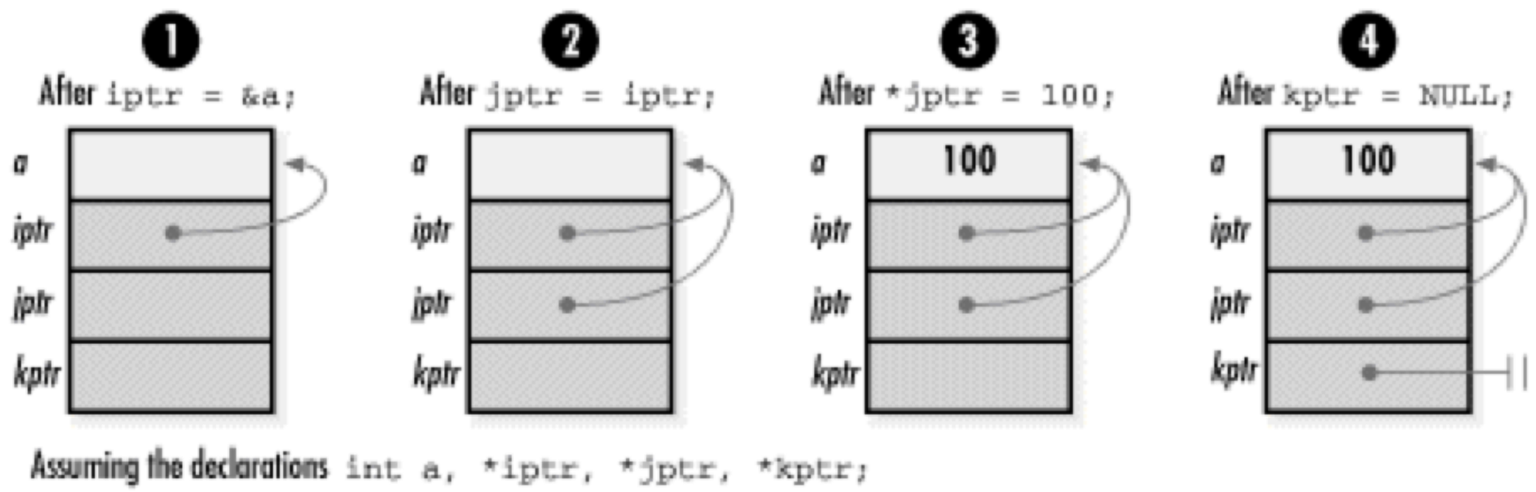
- What does it mean to declare a variable?

Pointer manipulation.

- For any type *T*, we can form a corresponding type for variables that contains addresses in memory where objects of type *T* resides.
- Look at them as like they are "pointing to" the objects. Therefore the name "Pointers".
- They are powerful means of building data structures and a way of precisely manipulating memory.
- But be careful! Misuse often leads to unpredictably buggy codes.

Pointer manipulation. Fundamentals

- A pointer is simply a variable that stores the address where a piece of data resides in memory rather than storing the data itself.
- When pointer points to nothing at all we set it to NULL.
- Pointers point to anywhere useful until we explicitly set it.
- Pointers the point to invalid addresses are sometimes called dangling pointers. For example, casting arbitrary integers to pointers, adjusting pointers beyond the bounds of arrays, and deallocating storage that one or more pointers still reference.



Pointer manipulation. Storage allocation

- When we declare a pointer in C, a certain amount of space is allocated for it. The same as for other types of variables.
- Pointers generally occupy one machine word, but their size can vary.
- Therefore never assume that a pointer has a specific size.
- Remember: when we declare a pointer, space is allocated only for the pointer itself. No space is allocated for the data the pointer references.
- Storage for the data is allocated in two ways: malloc or realloc.
- Automatic variables:
- Memory leaks. Blocks of storage that are allocated but never freed by a a program.

```
int f(int **iptr) {

int a = 10;
*iptr = &a;

return 0;

}
```


Pointer manipulation.
Aggregates and pointer arithmetic

```
typedef struct ListElmt_ {

void                *data;
struct ListElmt_    *next;

} ListElmt;
```

- Aggregates are structures and arrays. (Unions).
- Structures are not permitted to contain instances of themselves, but they may contain pointers to instances of themselves.
- Pointer arithmetic defines the rules by which calculations with pointers are performed.
- Pointers to structures are important in building data structures.
- Arrays and pointers in C use pointer arithmetic in the same way.
- *(a + i) equivalent to a[i].
- Remember that in C, multidimensional arrays are stored in row-major order.

Array Reference	Pointer Reference

`int f() {`
 	
`int a[10], *iptr;`
`iptr = a;`
`iptr[0] = 5;`
 	
`return 0;`
 	
`}`
	`int g() {`
	
	`int a[10], *iptr;`
	`iptr = a;`
	`*iptr = 5;`
	
	`return 0;`
	
	`}`

Conclusions: Pointers

We have seen:

- Data structure concept.
- Algorithms. Their efficiency. How they are closely related to data structures.
- Software engineering.
- Pointers manipulation to create data structures:
  a. Pointer fundamentals.
  b. Storage allocation.
  c. Aggregates and pointer arithmetics.

References and material

- Cosmología numérica y estadística: NagBody kit (http://bitbucket.org/rodriguezmeza). Mario A. Rodríguez-Meza. And: https://github.com/rodriguezmeza/NagBody_lectures.git

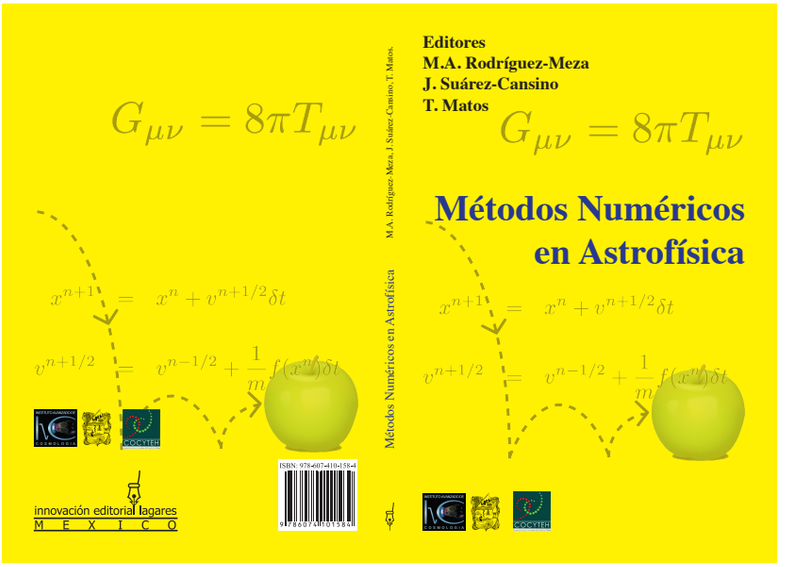
- Métodos numéricos en astrofísica, capítulo 1, Método de N-cuerpos en astrofísica. (https://www.researchgate.net/publication/316582859_Metodo_de_N-Cuerpos_en_Astrofisica)

- La estructura a gran escala del universo. Capítulo 22 en Travesuras cosmológicas de Einstein et al. https://www.researchgate.net/publication/316582400_La_estructura_a_gran_escala_del_universo_simulaciones_numericas

- https://www.researchgate.net/profile/Mario_Rodriguez-Meza

- https://www.researchgate.net/publication/314281416_Los_agujeros_negros_y_las_ondas_del_Dr_Einstein

- M.A. Rodriguez-Meza, Adv. Astron. 2012, 509682 (2012). arXiv: 1112.5201. (https://www.researchgate.net/publication/51967093_A_Scalar_Field_Dark_Matter_Model_and_Its_Role_in_the_Large-Scale_Structure_Formation_in_the_Universe)



See you!