**Pointers**

Let’s create a variable and print it :

 #include <stdio.h>

    int main (void){

    int var = 5 ;

    printf("Var : %i \n",var);

    }

If i ask where this variable is saved , you will tell me that it’s saved in memory , specificly in RAM (Random access memory ) as its name refer if’s save data randomly , and for prove to you , i will print the variable and the address where the ram saves them , two times

#include <stdio.h>

**1 Var : 5**

**address of var : 0x7ffece57f5dc**

**2 Var : 5**

**address of var : 0x7ffdc78d76ac**

    int main (void){

    int var = 5 ;

    printf("Var : %i \n",var);

    printf("address of var : %p \n",&var);

    }

AS the experience show , the first time Ram save the variable in an address but in the secnd it save it in diffrent place , in the experience we print the address of the variable so , we write **&variable-name** but the address is not and int value but it has a special formating wich is **: « %p »** **as note : address is writed by hexadecimal system**

But know we have another mission is saving the address in a variable

 #include <stdio.h>

    int main (void){

    int var = 5 ;

    int\* p = &var;

    printf("address of var : %p \n",p);

    }

Int\* 🡺 pointer refer to int value or refer to int variable we can also write : int \*p , with int \* is data type « contain adress hex «

P is a pointer contain address of int value . (the same thing for float ..)

The next mission is print the value that an address refer

#include <stdio.h>

    int main (void){

    int var = 50 ;

**Or write « \*&var«**

**the same thing**

    int\* p = &var;



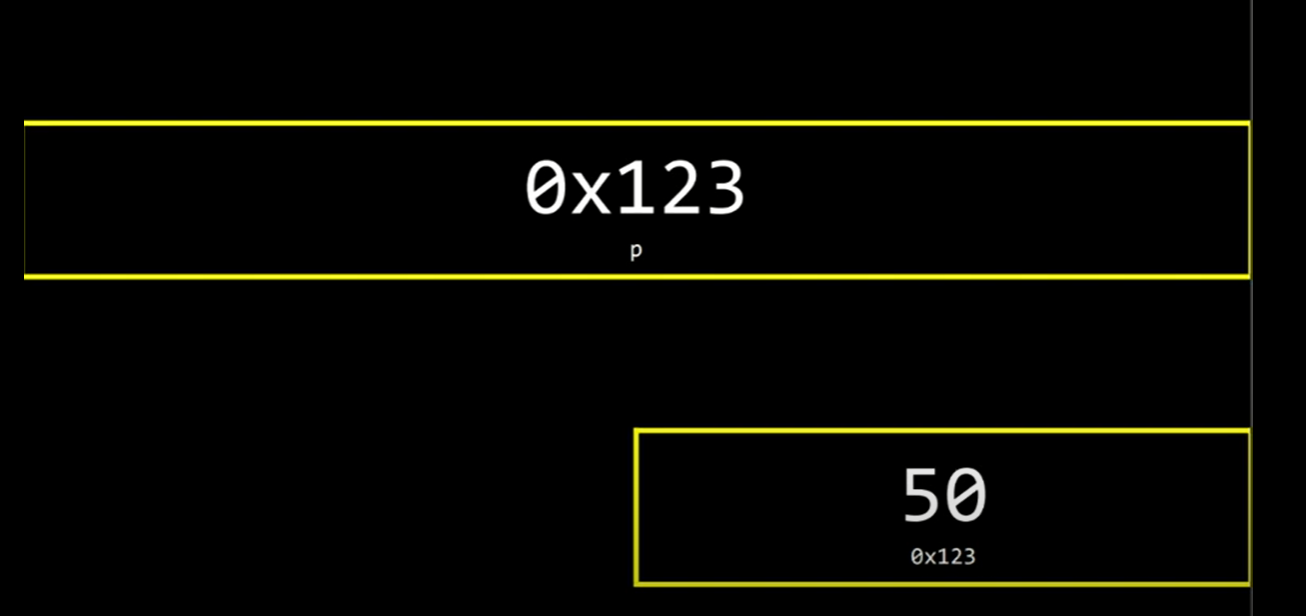
printf(" in ' %p ',  I find : %i \n",p,\*p);



    }

When we write \* before an address mean 🡺 the value that this pointer refer to it

Note : here we save the address of var in memory as we know 50 is integar so it will be saved in 4 byte , and the pointer will be save in

8 byte , this pointer refet to var ( tell us where is var )



String

As we know there is no data type named string in C language , and this type of data is made by cs50 and if you don ‘t call cs50 libary the code will not work ,and the Qu here is : « if we should use cs50 libarary in all our life «  sure NO

But language C has format %s and also has ‘‘…. ’’ , so to be honest

I will tell you That C language has format that work with string and it can work with text or string but there is no data type named string , so different in the name only .

So So As we know when we call cs50 library the code run as well

But let up know how this works :

**When you create a text string , all the caracter save in memory consecutively , or in the good order sure + null ‘\0’ which refer to the end of the text , but if you look as well here there is no string named ‘s’ here we have just sequence values , so to be honest the string s isn’t here abolutly but ‘s’ it’s a pointer wich refer to the first char in the text and for prove that we will print s as a pointer and see .**

#include <stdio.h>

Here we don’t write & before s because as we see the string in the reality is a pointer refer to the first char in the text .

But Here we write & because s[0] is a char but we add & for print it address

#include <cs50.h>

    int main (void){

    string s ="HI!";

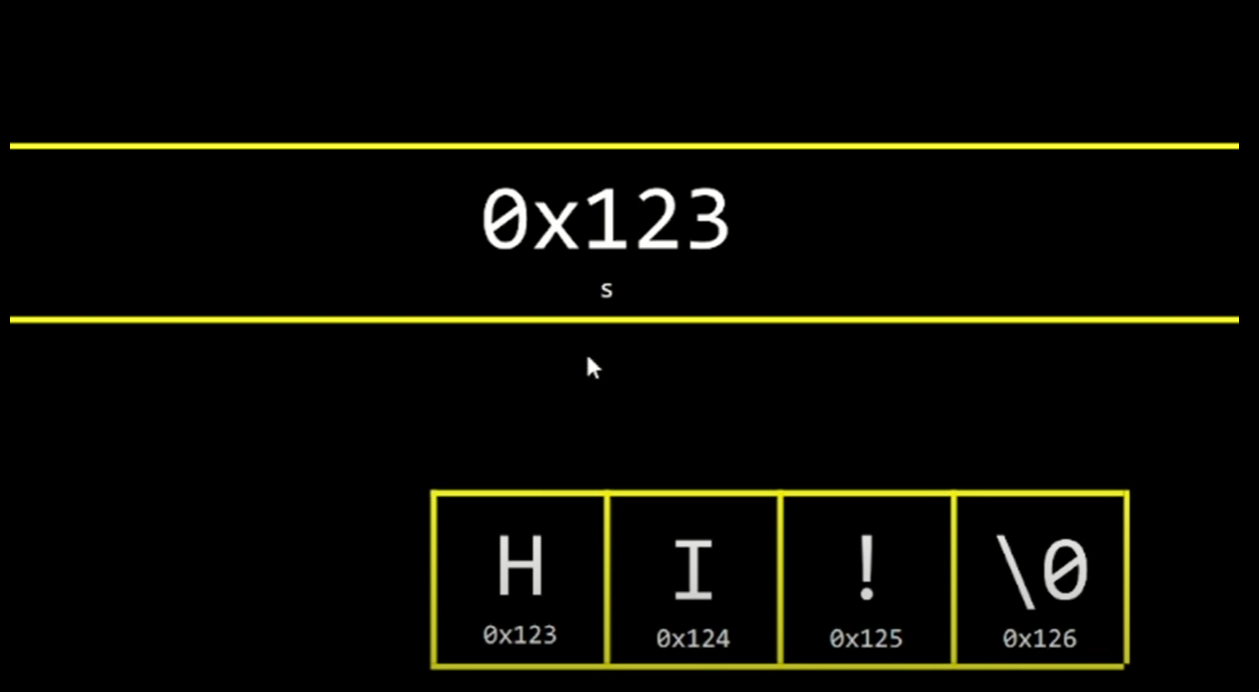
    printf("%p\n",s);

    printf("%p\n",&s[0]);

    }

**Result :**

**0x55a35f8dc004**

**0x55a35f8dc004**

 #include <stdio.h>

#include <cs50.h>

Result :

0x55dcc752b004

HI

    int main (void){

    string s ="HI!";

    printf("%p\n",s);

    printf("%c\n",\*s);

    printf("%c\n",\*s+1);

    }

**So string is a pointer refer to first char in the text and all values are in order ‘ when pointer refer to first char it continue the road to other value until it finds \0 so it read the first then second then thred .. until find null which refer it the end . but … ? stop here , in int before declare a pointer that refer to an int address «  int\* name «  but here why we don’t do that we should write «  char\* s «  because it refer to char value let test this .**

#include <stdio.h>

Result :

HI!

    int main (void){

    char\* s ="HI!";

    printf("%s\n",s);

    }

**Sttooooop : you tell me that we can only save in pointer adress not a string i will answear you and say that char is has particluar case , in char pointer you can save in it address normal but + string . The same thing that we say in string we will say it in char star \* char\* is a pointer refet to the first char in the text . so know this code will run without calling cs50.library .**

Why cs50 dosn’t tell us in the beggning of the course , because we will not understand it us well so she help us until learning pointer

#include <stdio.h>

    typedef char\* string;

    int main (void){

         string name="Bilal";

         printf("%s\n",name);

    }

So Here we do what css50 did , we remplce name « string «  by char pointer or char star \* , so know we can use string without calling cs50 and with the same name

#include <stdio.h>

    typedef char\* Bilal;

    int main (void){

         Bilal name="Bilal";

         printf("%s\n",name);

    }

So using typdef you can give other name to all know we can use char\* or we can call it Bilal ehh it work like string .

As we say all thing that we say it for string we say in on char\* because string is char\* with onther name

Char\* array of chras they are sequence values .

#include <stdio.h>

    int main (void){

         char\* s = "HI!";

         printf("%p\n",s);

         printf("%p\n",&s[0]);

    }

**Pointer Arithmetic**

Today we will gona learn how memory read the array or how the memory see the array , so let ‘s get started .

For well understanding we will create string variable and show its content lettrer by letter

#include <stdio.h>

Result :

H

I

!

The programm finish .

#include <cs50.h>



    int main (void){



        string s = "HI!";



        printf("%c\n", s[0]);

        printf("%c\n", s[1]);

        printf("%c\n", s[2]);

        printf("%c\n", s[3]);

    }

As you see we use here index for print each letter , if see and observe as well you will see that in index 3 as we know we have null and when we print it we get none (new line)

We said that string or char\* star is an address in memory that refer to the first letter in the text H , and as we know if we want to enter to this address and get its content we use \* , and we know that char\* is a pointer thar refer to the first letter in the array so for print ‘I’ we will enter to the next byte after H because in the array we have squence values consecutive

#include <stdio.h>

#include <cs50.h>

    int main (void){

        char\* s = "HI!";

        printf("%c\n", \*s);

        printf("%c\n", \*(s+1));

        printf("%c\n", \* (s+2));

        printf("%c\n", \*(s+3));

    }

But make in mind that this method is’n good , and we will know why.

**Note : Array is a pointer that refer to the first index in the array , when we write char\* or string and we say that char\* or string is a**

**Pointer because char\* or string is an array , so the origin is with the array and note with char\* or string . the origin is evry array is a pointer .**

For prove that :

#include <stdio.h>

#include <cs50.h>

Result :

0x7ffe7e9d28c0

0x7ffe7e9d28c0

    int main (void){

    int numbers[] = {5,2,8,9};

    printf("%p\n",numbers);

    printf("%p\n",&numbers[0]);

    }

AS you see the same address between array and the first array index .

#include <stdio.h>

 #include <cs50.h>

    int main (void){

    int numbers[] = {5,2,8,9};

    printf("%i\n",\*numbers);

    printf("%i\n", \*(numbers+1));

    printf("%i\n", \*(numbers+2));

    printf("%i\n", \*(numbers+3));

    }

This method isn’t good because :

As we know numbers is an address and \*numbers we enter to this address and when we add 1 we enter to the next byte and Here is the problem , are numbers saved in one byte ? sure answear is no because int take 4 byte , and here we have an intellegent compiler when you tell us to move to the next byte it moves to next base of data or next element (if you in 5 it will move to 2 ) , So this method isn’t good , the right method and the best method is using index .

#include <stdio.h>

 #include <cs50.h>

    int main (void){

    int numbers[] = {5,2,8,9};

    printf("%i\n", numbers[0]);

    printf("%i\n", numbers[1]);

    printf("%i\n", numbers[2]);

    printf("%i\n", numbers[3]);

    }

**Compare**

Now we have a mission , Mission says « You have to compare to values , if values are the same print same if not print Diffrent . «

 #include <stdio.h>

 #include <cs50.h>

    int main (void){

     int i = get\_int ("i : ");

     int j = get\_int("J :");

     if(i==j){

      printf("The Same \n");

     }

     else{

      printf("Diffrent \n");

    }

    }

This mission is very beautiful and very easy , but if Know instead of int we have string or char \*.

 #include <stdio.h>

 #include <cs50.h>

    int main (void){

     string A = get\_string("A: ");

     string B = get\_string("B: ");

     if(A==B){

      printf("The same \n");

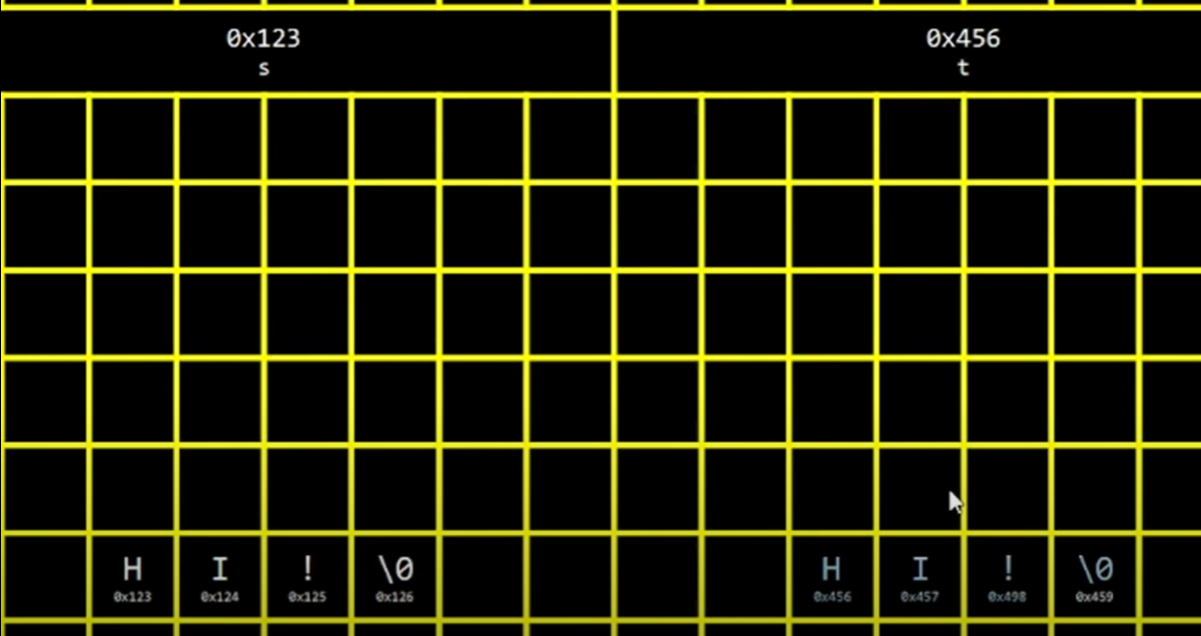
     }

     else{

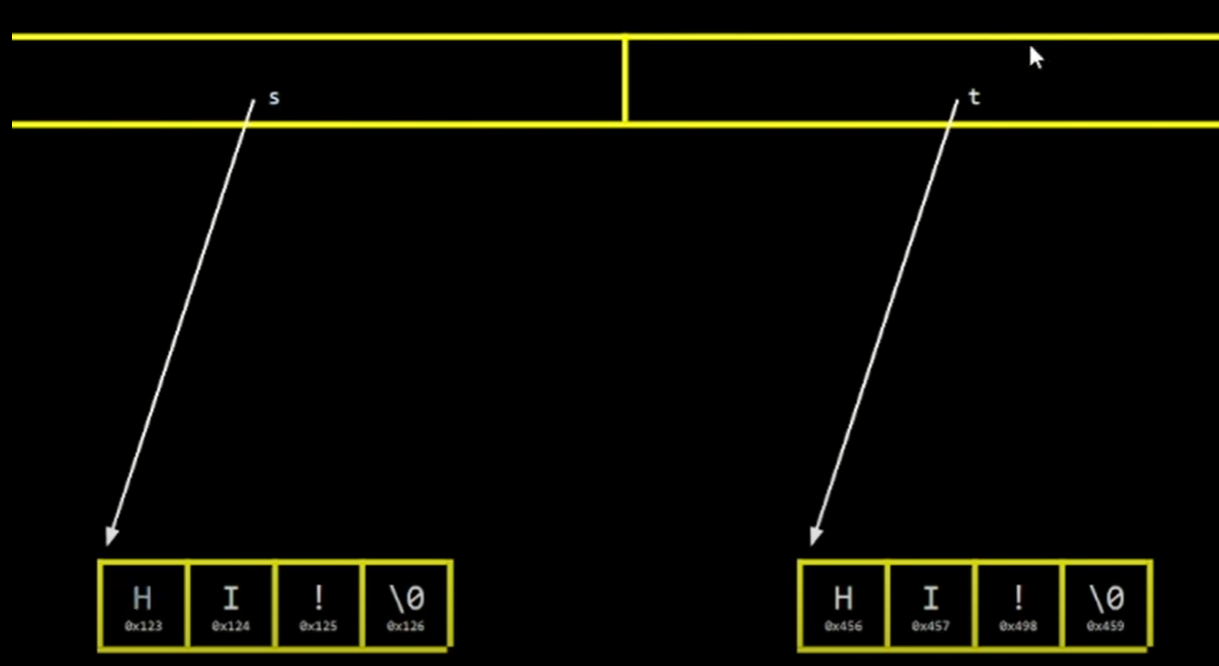
      printf("diffrent \n");

     }

    }

This is the programm « hhhh » . but if we test , always we will get diffrent , But why The ANSWEAR : string or char\* is a pointer contain the address of the first index in the array so when you compare A and B you comare two addresses two hexadecimal and sure you will never get same bacause we can’t save many data in one place

As the img show when you compare two string , you just compare the adrress of the first index in arrays



And for well prove that by code :

#include <stdio.h>

 #include <cs50.h>

    int main (void){

**Result :**

**A: Bilal**

**B: Bilal**

**A : 0x55a032bfe6b0 , B : 0x55a032bfe6f0**

**diffrent**

     string A = get\_string("A: ");

     string B = get\_string("B: ");

     printf("A : %p , B : %p \n",A,B);

     if(A==B){

      printf("The same \n");

     }

     else{

      printf("diffrent \n");

     }

    }

And the solution of this mission is using library :

#include <stdio.h>

 #include <cs50.h>

 #include <string.h>

    int main (void){

     string A = get\_string("A: ");

     string B = get\_string("B: ");

     if( strcmp(A,B)==0 ){

      printf("The same \n");

     }

     else{

      printf("diffrent \n");

     }

    }

Sartcompare or stcmp from string.h library help us to compare two strings . (The case of letters is sensitive )

Or use startcasecompare or stcasecmp from strings.h library help us to compare two strings (ignoring The case of letters )

Hear Hear it sound like a clever man speak , yes he says i have another solution :

#include <stdio.h>

 #include <cs50.h>

    int main (void){

     string A = get\_string("A: ");

     string B = get\_string("B: ");



     if( \*A == \*B){

      printf("The same \n");

     }

     else{

      printf("diffrent \n");

     }

    }

Hhhhh no he is not smart and this isn’t right , because you compare the first indexes in the arrays only not all the text .

**Malloc & free & valgrind**

So Here we will learn Manual memory control , we will learn how to reseve a place in memory and how to make it free .

For Well understanding , Know we will create a variable and bring its value from user , then copy its value and put it in other variable , such as when we modify the copy the origin stay as it is . (modify not change)

#include <stdio.h>

 #include <cs50.h>

Result :

S : bimad

s : Aimad || t : Aimad

    int main (void){

      char\* s = get\_string(" S : ");

      char\* t = s ;

      t[0]='A';

      printf("s : %s  || t %s \n", s ,t);

    }

As you see , so this method isn’t right , and we don’t do a copy because the origin change too , let’t do cs50 origin example

#include <stdio.h>

 #include <cs50.h>

 #include <ctype.h>

**Result :**

**S : bilal**

**S : Bilal || T : Bilal**

    int main (void){

      char\* s = get\_string(" S : ");

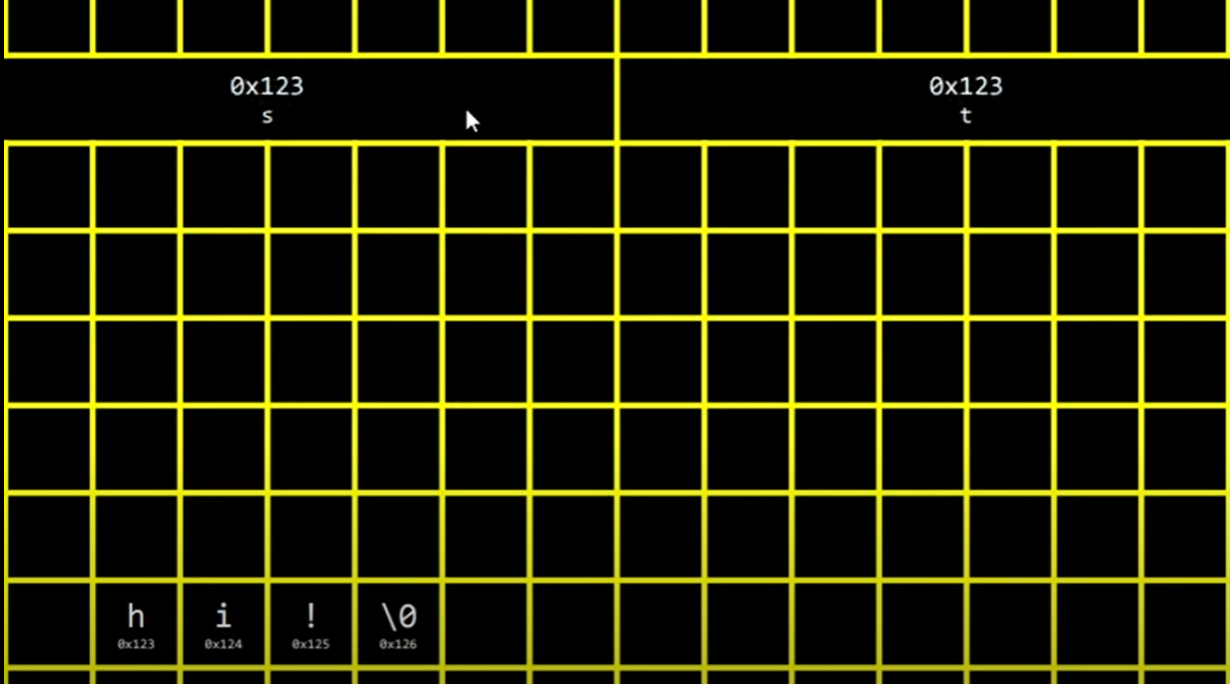
      char\* t = s ;

      t[0]=toupper(t[0]);

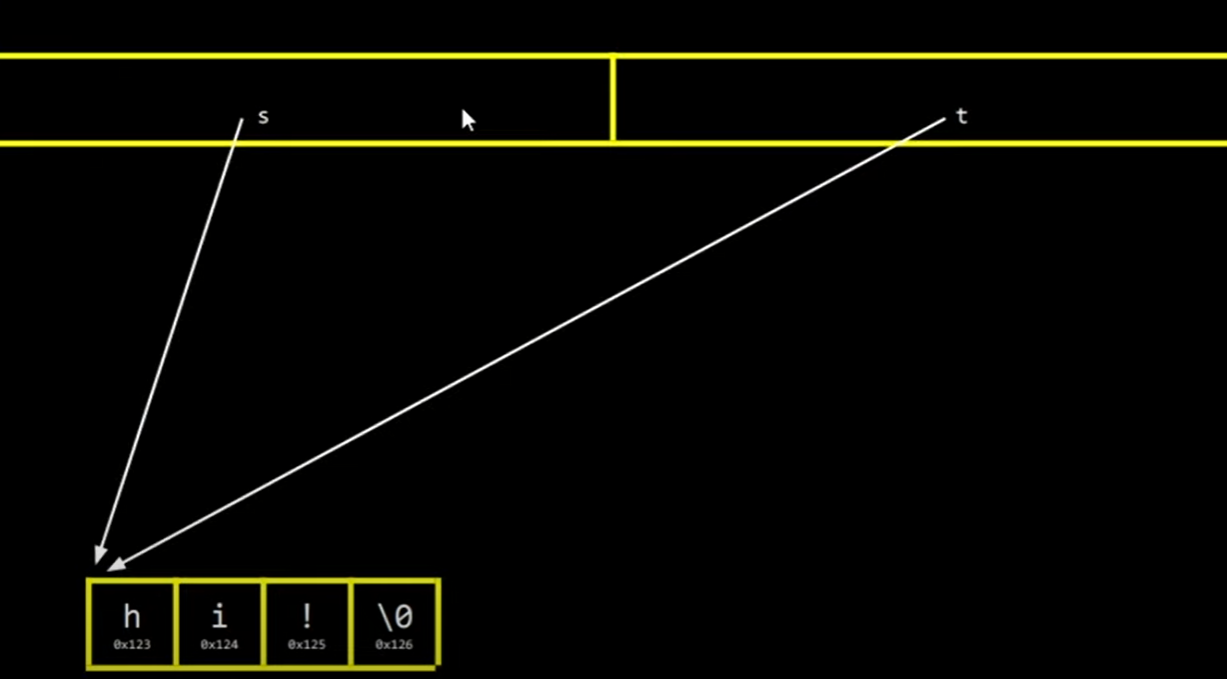
      printf(" S : %s  || T : %s \n", s ,t);

    }

Here Also we change The copy , but the origin also change , so we didn’t make a copy , but we put the origin in t and the modification made on the origin , But Why ?

Focus with me here s is a pointer , is a hexadicimal (address) that refer to the first index , and also char\* t is a pointer , when we write t=s so two variables will refer to the same index .because char\*t take the addres from char\* s :

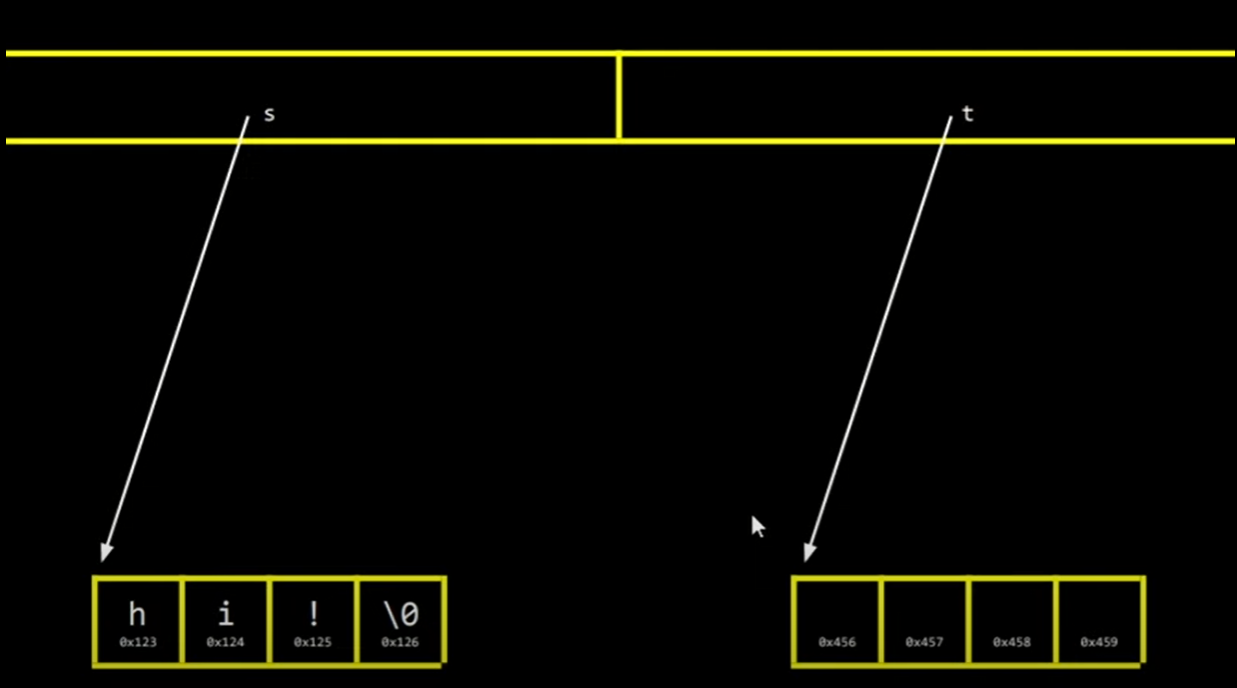
As you see s and t become pointers refer to the same index , so when you do a modification in any one of them , the other will receive also the modification , because they are the same .

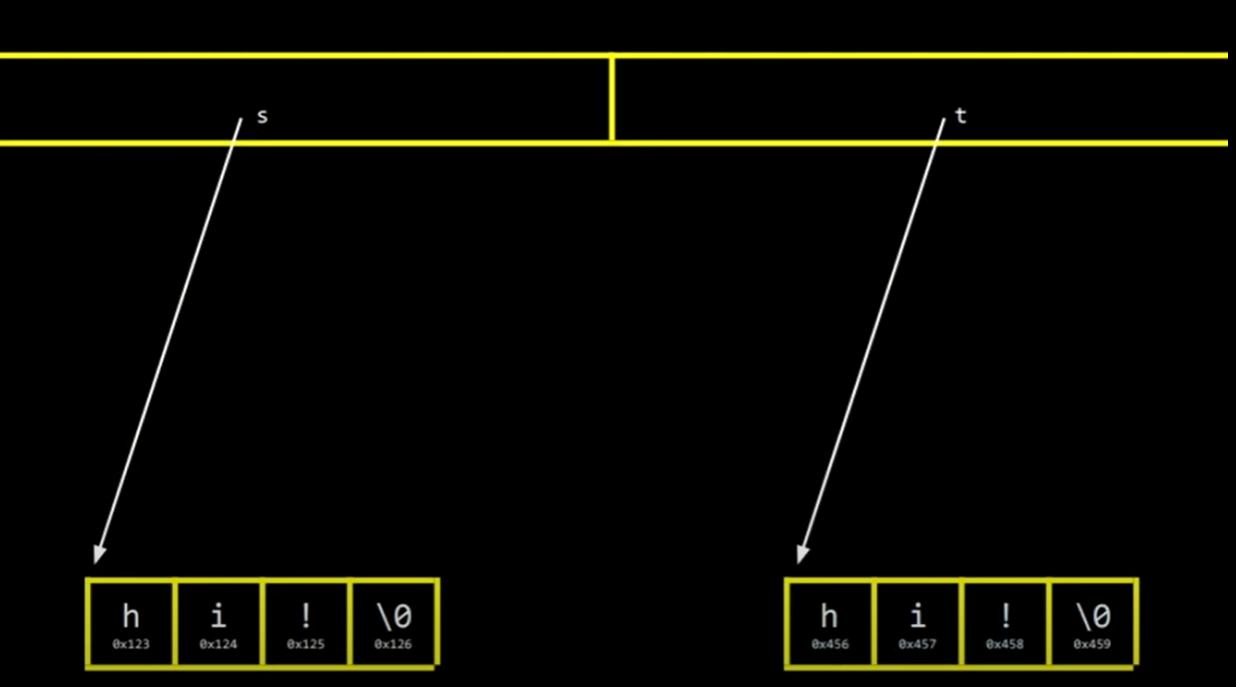


And the solution is reserve a place in memory then make t refer to the first index in the reseved place then start transfering the letters .

1. **Take a place in momery .**



1. **make t refer to the first index in reseved place** :

Then : **3 . transfering the text from s to t**

Know we will do that by code :

First thing we will not make char\* t point the first index of char\* s , if we do that we will break all , so first thing we will reseve a place in memory and make t point to it .

For do that we will use malloc function from stdlib.h .

Syntax : malloc(byte) ;

#include <stdio.h>

 #include <cs50.h>

 #include <string.h>

 #include <stdlib.h>

    int main (void){

      char\* s = get\_string(" S : ");

      char\* t = malloc(strlen(s)+1) ;

}

We can’t know how much letter that the user will enter but here we used strlen that retrun number wich indicate how much there are letters , we add one for NULL \0 .

And know we will transefer the letters using loop :

#include <stdio.h>

 #include <cs50.h>

 #include <ctype.h>

 #include <string.h>

 #include <stdlib.h>

    int main (void){

      char\* s = get\_string(" S : ");

      char\* t = malloc(strlen(s)+1) ;

      for (int i=0 ; i<strlen(s)+1;i++){

         t[i]=s[i];

      }

      t[0]= toupper(t[0]);

      printf(" S : %s  || T : %s \n", s ,t);

    }

I have onther method using startcopy or strcpy for string.h .

Strcpy copy a string , including its terminating ‘\0’ to dest ‘متلقي’

#include <string.h>

char \*strcpy(char \*dest, char \*src);

dest : destination , src : source ,

Note : Know we reseved memory manualy , and we used malloc so there are many things that we must be carfull from it .

Malloc(4) ; know we request to operation sytem to bring 4 byte from memory , by example os doesn’t find enought place in memory , so malloc will return null , and if t =null and you continue the program so here you have a problem , so for avoid that :

 #include <stdio.h>

 #include <cs50.h>

 #include <ctype.h>

 #include <string.h>

 #include <stdlib.h>  
    int main (void){

      char\* s = get\_string(" S : ");

      char\* t = malloc(strlen(s)+1) ;

      if(t==NULL){

         printf("There is no space to run the program");

         return 1 ;

      }

      for (int i=0 ; i<strlen(s)+1;i++){

         t[i]=s[i];

      }

      t[0]= toupper(t[0]);

      printf(" S : %s  || T : %s \n", s ,t);

    }

Or inseted of « if == null «  you can write all programm in « if t != null «

When you do int x = 10 ; here automaticly reseve place in memory then make it free , but here you reserve manualy so you have to make it free manualy also .

 #include <stdio.h>

 #include <cs50.h>

 #include <ctype.h>

 #include <string.h>

 #include <stdlib.h>

    int main (void){

      char\* s = get\_string(" S : ");

      char\* t = malloc(strlen(s)+1) ;

      if(t==NULL){

         printf("There is no space to run the program");

         return 1 ;

      }

      for (int i=0 ; i<strlen(s)+1;i++){

         t[i]=s[i];

      }

      t[0]= toupper(t[0]);

      printf(" S : %s  || T : %s \n", s ,t);

      free(t);

    }

We use free function from stdlib.h , if you don’t free the memory that you bring it you will have a problem in memory .

For know if we have a problem in memory we use valgring

$ make test

$ valgrind ./test

If you don’t know how to read the result use help50

$ help50 valgrind ./test

**Swap**

What is sawp : for well inderstanding swap

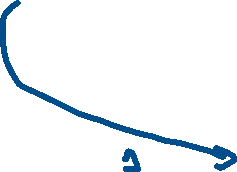
Int x = 1 ;

Int y = 2 ;

Swap mean swith between variables , which mean x will equal 2 and y will equal 1 .

For swap or switch between variables , we need other variable , for put in it one variable and then we can swap like this cups example .





Let’s see that with code :

#include <stdio.h>

 #include <cs50.h>

    int main (void){

    int x = 1 ;

    int y = 2 ;

     int temp = x ;

     x=y;

     y=temp;

    printf("x : %i \n",x);

    printf("y : %i \n",y);

    }

But not this that I want to do , i wana do fonction that i give it x and y and it swaps them .

 #include <stdio.h>

 #include <cs50.h>

void swap (int a , int b);

    int main (void){

    int x = 1 ;

    int y = 2 ;

    swap(x,y);

    printf("x : %i \n",x);

x : 1

y : 2

    printf("y : %i \n",y);

    }

 void swap (int a , int  b){

   int  temp = a ;

   a=b;

   b=temp;

 }

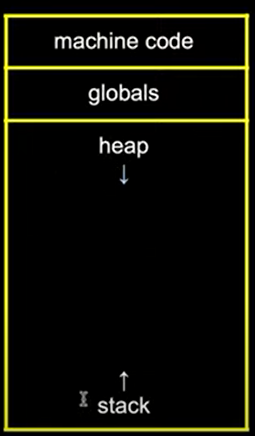
As you see The varibles still as they are , no swap but why :

This function that you do , swap a and b , not x and y ????

For well inderstanding This , we will see how memory read this code .

Is this code saved randomly in the memory , answear is NO

The memory divided to many place . each place in memory save a part of you program

**machine code**: contain 0 and 1 , when we just open a program all 0 and 1 goo to save here in this part .

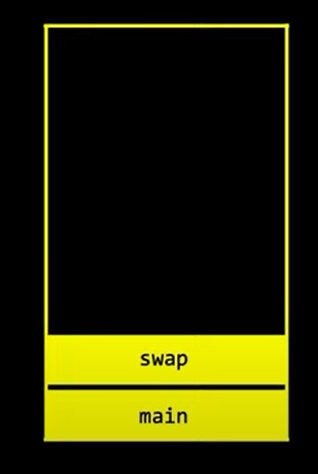
**globals :** global variables are saved Here .

every variable created in a function named local variable , and every varible created out of function named global variable which mean above of main fuction and not in any other function , we can access to any global variable from any function

**heap :** malloc() and free() , evry thing saved here and freed from here made manualy as you see we have down arrow which mean

heap fill from top to bottom . when you make many malloc heap will fill and also there is a problem .

**stack :** every function or local variable is saved here and all thing saved here or freed from here made automaticly , as the arrow indicate stack fill from bottom to top . if you do many local variables the stack will fill and sure when stack fill this is a problem and this problem name stackoverflow .

*so when you start your programm the first thing that will start is main function when os enter main function os will find a call to swap function so , os will start swap and find that you swap a and b and not x and y ;*

**NOW FOCUS HERE :**

**When you give a value in a fuction , the argument receive a copy from variable value and not the reference which mean you take pass by value and not pass by refrence**

**Pass By refrence :** we see it in malloc episode , there we put s pointer in other variable and when we do some modifiction on other variable we applied also on the refrence (because two of them have the same pointer in memory ) and this is named pass by refrence . you give the pointer of an array to anoher variable , and here you give the origin or the reference .

**Pass By value :** Here you send a value to the function By example : swap function , that named pass value because you send the value (not the address in memory ) , Here you take the value of X variable and send it to varible a in swap function , so here when you modify a in swap function you modify the copy not the refrence (because you take its value not it address in memory so any modification on ‘a’, x will not receive anything , this named pass by value or pass by copy .

 #include <stdio.h>

 #include <cs50.h>

void swap (int\* a , int\* b);

    int main (void){

    int x = 1 ;

    int y = 2 ;

    swap(&x,&y);

    printf("x : %i \n",x);

    printf("y : %i \n",y);

    }

 void swap (int\* a , int\*  b){

   int  temp = \*a ;

   \*a=\*b;

   \*b=temp;

}

*So Know we do pass by refrence so we modify on refrence because we send an address in memory so we modify a specify variable in memory .*

*And as we say stack start automaticly he also delete automacily so it the opposite of heap .*

**Scanf**

In this episod we will learn how to dispense on cs50 library , and we will learn how to get data from user ,

**User Input :** You have already learned that printf() is used to output values in C

To get ***user input*** , you can use the Scanf() function :

// Create an integer variable that will store the number we get from the user  
int myNum;  
  
// Ask the user to type a number  
printf("Type a number: \n");  
  
// Get and save the number the user types  
**scanf**("%d", &myNum);  
  
// Output the number the user typed  
printf("Your number is: %i", myNum);

**The scanf() function takes two arguments :**

1. *Format specifier of the variable (%i in the example above)*
2. *The reference operator (&myNum) , which strores the memory address of the variable .*

**User Input char\* or text :**

#include <stdio.h>

 int main (void){

   char\* s;

   printf("Type you first name  : \n ");

   scanf("%s",s);

   printf(" your first name is  %s\n",s);

 }

But if we run this program we will get an error , because here you tell the os to store the input user in s , and s in a pointer that don’t point or don’t indicate anything because you declare it and close it .

So char\* don’t point on anything in memory and there is no reseved place to save in it , when you try to compile programm you will get error and a message tell you that char\* s == NULL , which mean char\* s point on nothing

For slove this problem you have to make the pointer point on a place in memory that you reserve it .

1-using malloc .

#include <stdio.h>

 #include <stdlib.h>

 int main (void){

Here we don’t write reference operetor « & » because s is a pointer or it an adress

   char\* s=malloc(10);

   printf("Type you first name  : \n ");

   scanf("%s",s);

   printf(" your first name is  %s\n",s);

     free(s);

 }

2- using char array

#include <stdio.h>

 int main (void){

   char s [15];

   printf("Type you first name  : \n ");

   scanf("%s",s);

   printf(" your first name is  %s\n",s);

 }

And sure this method also save or reseve a place in memory .

The problem is that if user enter more than letters that i reseve place for , here scanf try to slove the problem so we can get no error but in the same time we can get it , so it not secured

We must specfy the size of the array , and reserve the maximum logic value , in example above at maximum first name will have 15 letters .

**Note : %% mean the % symbole .**

**File In c**

So If i have an external file , how to work with it using c ??

So for open a file we use fopen() function , this function take 2 argument string , the first one is the name of the file that you want work on , the second one is mood .

//   mood                Delete         create

//  r => read             no              no

//  w => write            yes             yes

//  a => apend            no              yes

Now we will learn 3 moods (main moods) :

R , read 🡺 opens an existing file .

W , write 🡺 opens a text file for writing , if the file doesn’t exist then a new file is created , if the file is exists then make it blank .

A , append 🡺 opens a text file for appending (modify) wich mean writing at the end of an existing file , and create the file if doesn’t exist .

**Return Value**

Fopen () function returns NULL in case of a failure , and retruns File pointer (File address in memory ) on success .

As we say fopen returns pointer so we have to save it in pointer . but which data type emmm ?? answear is a new data type named FILE , so it will become FILE\* (pointer) .

**New Mission :** phonebook , get name and number phone from user then write this in the file .

For adding data in a file , we use fprintf() function , this function take two string the first one is the name of the file and the second one is message that you want you print in the file .

Note : we open a file so we must close it , and for close it we use , fclose() function , but why i have to close the file ???

So The answear is : «  all the code that you write is stored in Ram and for save this in hard desk we use fclose()

#include <stdio.h>

 #include <cs50.h>

 int main (void){

      FILE\* myfile = fopen("phonebook.csv","a");

      char\* name = get\_string("name : ");

      char\* phone = get\_string("number : ");

      fprintf(myfile , "Name : %s   ||   Number : %s \n",name,phone);

      fclose(myfile);

      printf("Congratulation you have created a new account \n");

 }

***Here we use a , mood because we want the programm write a the end of the file , for save the previous data***

**New Mission :** you have to make a programm that detect jpeg or jpg and png format .

So For That we have installed some images with diffrent format for test with it .

As we know every image is a group of pixels and each pixel is a color , the color is 3 bytes , each bytes is a hexadecimal number,

Evrey jpeg img it first three bytes are : 0xff 0xd8 0xff

And evry png img it first three bytes 0x89 0x50 0x4e

So know i want to enter to the img for know is it jpeg or not , and for make check , so i will use fopen() and i will you r mood because i will read the img info , but if i write the name of the img by myself so here the programm it not dynamique , but me i want the user who specify the name of the img , so i want to send the name to main function so i will sure recieve by an argument , and know we have to save ourselves againest the user , so the first thing , we will make sure that the user will enter one file name , and the second thing that this file name will be exist . so know we will start .

Know i want to enter the file and read the first 3 bytes then store them in an array . the type of the array will be char because char store in one byte and as we say we want to store in it the firt 3 bytes , but char has a problem , ff equal 255 and char grant us 255 but form -128 to 127 , and here we have to learn about signed and unsigned char,

Signed vs unsigned

So know we will tell the computer that we want to store an integer that mean that we want to reserve a place that we can’t store in it positive and negative values , so the computer reserve a place a place in memory that we can put in it at max 2147483647 for positive and a place for nagative that i can put in within -2147483647 and if i put more than this I will get error because for computer i am adding numbers more than computer store , and Here unsigned .

When i write unsigned before int , here the computer instead of it devide the place to two places now it reserve one place but all positive and feature here that i can fill it the double « 4294967295 »

So in our project we will use unsigned char for get all space positve for make it possible to arrive 255

And now i want to enter to the img and bring the first three byte

So here we will use fread() function this function take 4 argumts

The first one is the place where you will store in it the readed content

The second one mean how much i want store of bytes , choose one if it is bool or char and choose 4 for int ….

The third how much you will take this data

The fourth one is the the source or where i will bring all that or the fille that i will read it .

#include <stdio.h>

 #include <cs50.h>

 int main (int argc ,char\* argv[]){

     if(argc != 2 ){

          printf("You have To Enter one img name  , Try again \n");

          return 1 ;

     }

          FILE\* file = fopen(argv[1],"r");

          if (file == NULL){

               printf("There is no File With This Name \n");

               return 1;

          }

            unsigned char  c[3];

          fread(c,1,3,file);

          if (c[0]==0xFF && c[1]==0xD8 && c[2]==0xFF ){

               printf("Its A JPEG / JPG Img \n");

          }

          else if (c[0]==0x89 && c[1]==0x50 && c[2]==0x4e ){

               printf(" It's A PNG Img \n");

        }

         else{

                printf("I Can't Detect The Format Of This Img \n");

        }

          fclose(file);

 }