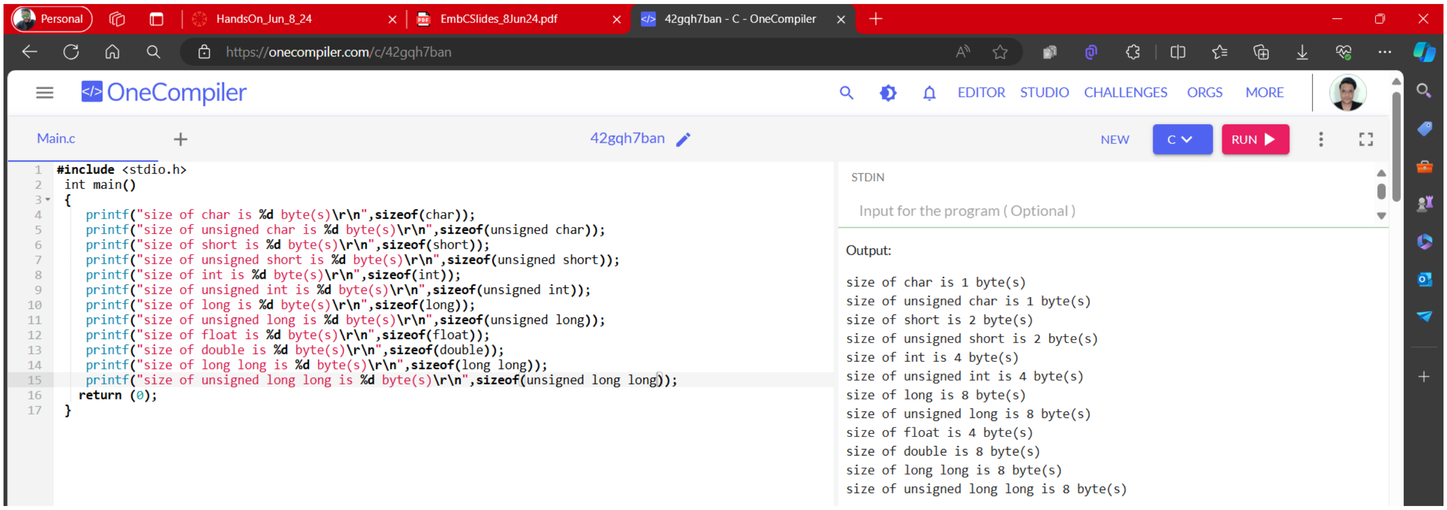
HandsOn: The memory size of C objects

HandsOn: The memory size of C objects

Check the size of the data types

- Test the following code
  - Run the code at the following web site, website2
- Add the lines to print all of the primitive and derived data types: char, unsigned char, short, unsigned short, int, unsigned int, long, unsigned long, float, double, long long, unsigned long long.
- Report code and image of the result

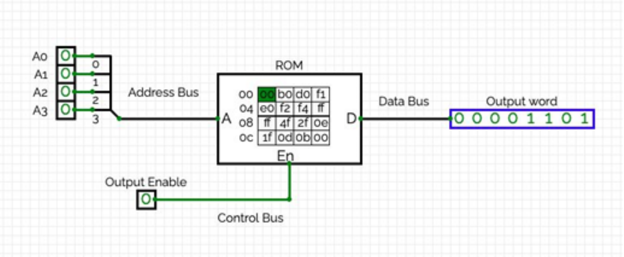


```c
#include <stdio.h>
 int main()
 {
    printf("size of char is %d byte(s)\r\n",sizeof(char));
    printf("size of unsigned char is %d byte(s)\r\n",sizeof(unsigned char));
    printf("size of short is %d byte(s)\r\n",sizeof(short));
    printf("size of unsigned short is %d byte(s)\r\n",sizeof(unsigned short));
    printf("size of int is %d byte(s)\r\n",sizeof(int));
    printf("size of unsigned int is %d byte(s)\r\n",sizeof(unsigned int));
    printf("size of long is %d byte(s)\r\n",sizeof(long));
    printf("size of unsigned long is %d byte(s)\r\n",sizeof(unsigned long));
    printf("size of float is %d byte(s)\r\n",sizeof(float));
    printf("size of double is %d byte(s)\r\n",sizeof(double));
    printf("size of long long is %d byte(s)\r\n",sizeof(long long));
    printf("size of unsigned long long is %d byte(s)\r\n",sizeof(unsigned long long));
   return (0);
 }
```

```
size of char is 1 byte(s)
size of unsigned char is 1 byte(s)
size of short is 2 byte(s)
size of unsigned short is 2 byte(s)
size of int is 4 byte(s)
size of unsigned int is 4 byte(s)
size of long is 8 byte(s)
size of unsigned long is 8 byte(s)
size of float is 4 byte(s)
size of double is 8 byte(s)
size of long long is 8 byte(s)
size of unsigned long long is 8 byte(s)
```

HandOn - Reading memory (ROM - Read Only Mem)

- Import to circuitverse the following circuit (link)
- Answer the following questions:
  - What is the ROM capacity?
    - How many memory locations?
    - What is the location (word) size?
  - How many address lines (bus) to access the ROM (addressing space)?
  - How many data lines (bus) for this ROM?
  - How many control lines (bus)?
  - What is the purpose of the output enable?
  - What data is stored into at memory location (08)$_{16}$?
  - What data is stored into at memory location (0A)$_{16}$?



00	00	b0	d0	f1
04	e0	f2	f4	ff
08	ff	4f	2f	0e
0c	1f	0d	0b	00

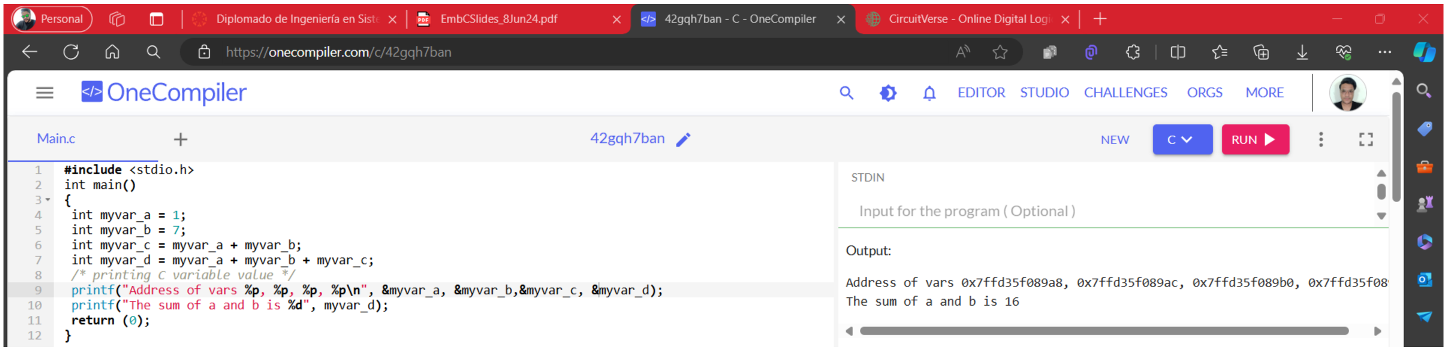
1. 128 bits
   a. 16 locations
   b. 8 bits
2. 4
3. 1
4. Shows the value on the current memory location when turned on, if turned off, keep as output the last value displayed.
5. 0x1F
6. 0x2F

HandsOn: Variables

Check the size of the data types for different architectures

- Test the following code
- Modify the code to perform myvard = myvara+myvarb+myvarc
- Modify the code to display the address of myvard
- Report code and image of the result

```
#include <stdio.h>
int main()
{
 int myvar_a = 1;
 int myvar_b = 7;
 int myvar_c = myvar_a + myvar_b;
 /* printing C variable value */
 printf("Address of vars %p,%p,%p", &myvar_a, &myvar_b,&myvar_c);
 printf("The sum of a and b is %d", myvar_c);
 return (0);
}
```


Personal
Diplomado de Ingeniería en Sist
EmbCSlides_8Jun24.pdf
42gqh7ban - C - OneCompiler
CircuitVerse - Online Digital Logi
https://onecompiler.com/c/42gqh7ban
OneCompiler
EDITOR
STUDIO
CHALLENGES
ORGS
MORE
Main.c
42gqh7ban
NEW
C
RUN
STDIN
Input for the program (Optional)

```
#include <stdio.h>
int main()
{
 int myvar_a = 1;
 int myvar_b = 7;
 int myvar_c = myvar_a + myvar_b;
 int myvar_d = myvar_a + myvar_b + myvar_c;
 /* printing C variable value */
 printf("Address of vars %p, %p, %p, %p\n", &myvar_a, &myvar_b,&myvar_c, &myvar_d);
 printf("The sum of a and b is %d", myvar_d);
 return (0);
}
```

Output:

```
Address of vars 0x7ffd35f089a8, 0x7ffd35f089ac, 0x7ffd35f089b0, 0x7ffd35f08
The sum of a and b is 16
```

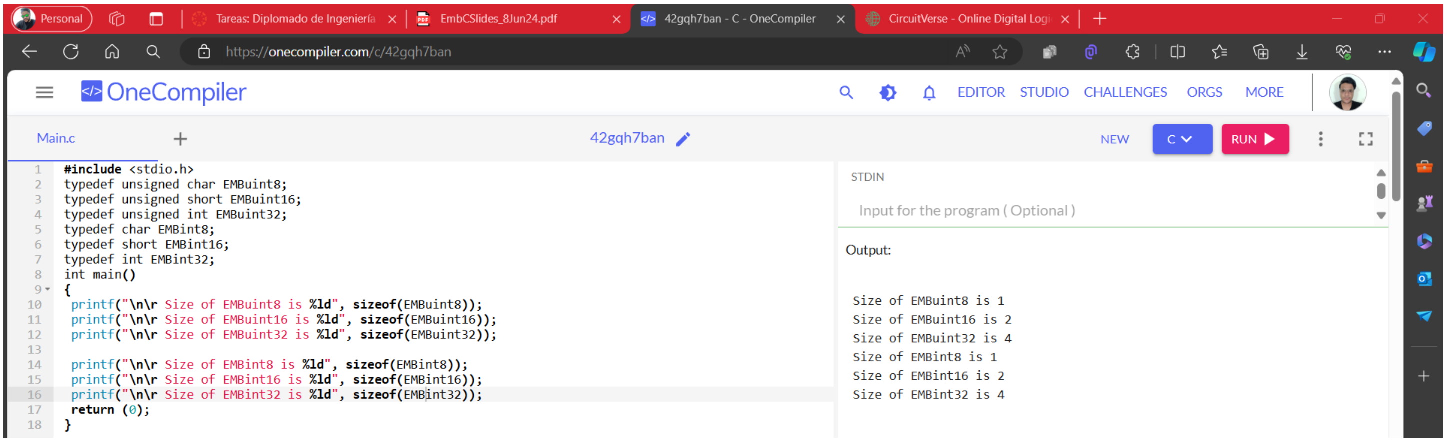
HandsOn: Typedefs

Check the size of the data types for different architectures

- Test the following code
- Write your own definition of the following types. Also, write the printf statement to print the size in bytes of each of previously defined types.
  - EMBuint8, EMBuint16, EMBuint32, EMBint8, EMBint16, EMBint32
- Report code and image of the result

```c
#include <stdio.h>
typedef unsigned char EMBuint8;
int main()
{
 printf("\n\r Size of EMBuint8 is %ld", sizeof(EMBuint8));
 return (0);
}
```

@ Tecnológico de Monterrey


Personal
Tareas: Diplomado de Ingeniería
EmbCSlides_8Jun24.pdf
42gqh7ban - C - OneCompiler
CircuitVerse - Online Digital Logi
https://onecompiler.com/c/42gqh7ban
OneCompiler
EDITOR
STUDIO
CHALLENGES
ORGS
MORE
Main.c
42gqh7ban
NEW
C
RUN
STDIN
Input for the program (Optional)
Output:

```c
#include <stdio.h>
typedef unsigned char EMBuint8;
typedef unsigned short EMBuint16;
typedef unsigned int EMBuint32;
typedef char EMBint8;
typedef short EMBint16;
typedef int EMBint32;
int main()
{
 printf("\n\r Size of EMBuint8 is %ld", sizeof(EMBuint8));
 printf("\n\r Size of EMBuint16 is %ld", sizeof(EMBuint16));
 printf("\n\r Size of EMBuint32 is %ld", sizeof(EMBuint32));

 printf("\n\r Size of EMBint8 is %ld", sizeof(EMBint8));
 printf("\n\r Size of EMBint16 is %ld", sizeof(EMBint16));
 printf("\n\r Size of EMBint32 is %ld", sizeof(EMBint32));
 return (0);
}
```

```
Size of EMBuint8 is 1
Size of EMBuint16 is 2
Size of EMBuint32 is 4
Size of EMBint8 is 1
Size of EMBint16 is 2
Size of EMBint32 is 4
```

HandsOn: Variables fixed width

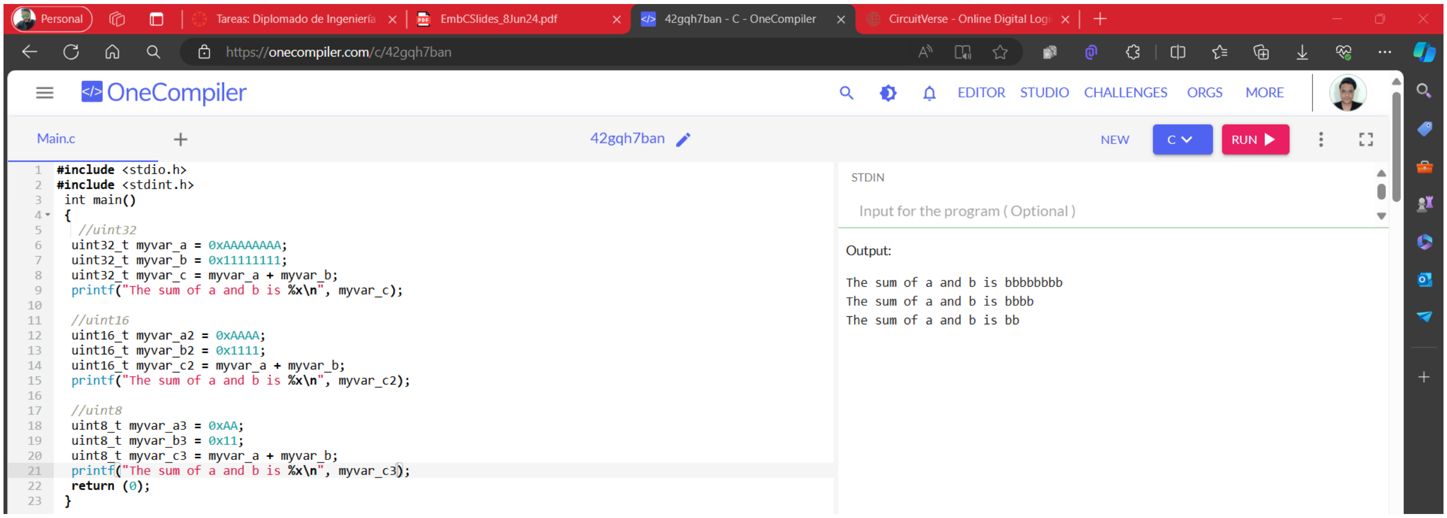
Check the size of the data types for different architectures

- Test the following code
  a) Run the code and verify that the correct result appears on memory
  b) copy/paste the code section and modify/run the code so the result is 12
  c) copy/paste the code section and modify the code to have "unsigned int" variables and initialize the variables myvar_a and myvar_b with the values 0xAAAAAAAA and 0x11111111 run the code and verify the result is correct
  d) copy/paste the code section and modify the code to have "unsigned short" variables. Using as reference the same initialization values from (c), initialize the variables and verify the result.
  e) copy/paste the code section and modify the code to have "unsigned char" variables. Using as reference the same initialization values from (c), initialize the variables and verify the result.
- Report code and image of the result

```
#include <stdio.h>
int main()
{
 int myvar_a = 1;
 int myvar_b = 7;
 int myvar_c = myvar_a + myvar_b;

 printf("The sum of a and b is %x", myvar_c);
 return (0);
}
```

@ Tecnológico de Monterrey



Personal
Tareas: Diplomado de Ingeniería
EmbCSlides_8Jun24.pdf
42gqh7ban - C - OneCompiler
CircuitVerse - Online Digital Logi
https://onecompiler.com/c/42gqh7ban
OneCompiler
EDITOR
STUDIO
CHALLENGES
ORGS
MORE
Main.c
42gqh7ban
NEW
C
RUN
STDIN
Input for the program (Optional)

```
#include <stdio.h>
#include <stdint.h>
 int main()
 {
  //uint32
  uint32_t myvar_a = 0xAAAAAAAA;
  uint32_t myvar_b = 0x11111111;
  uint32_t myvar_c = myvar_a + myvar_b;
  printf("The sum of a and b is %x\n", myvar_c);

  //uint16
  uint16_t myvar_a2 = 0xAAAA;
  uint16_t myvar_b2 = 0x1111;
  uint16_t myvar_c2 = myvar_a + myvar_b;
  printf("The sum of a and b is %x\n", myvar_c2);

  //uint8
  uint8_t myvar_a3 = 0xAA;
  uint8_t myvar_b3 = 0x11;
  uint8_t myvar_c3 = myvar_a + myvar_b;
  printf("The sum of a and b is %x\n", myvar_c3);
  return (0);
 }
```

Output:

```
The sum of a and b is bbbbbbbb
The sum of a and b is bbbb
The sum of a and b is bb
```

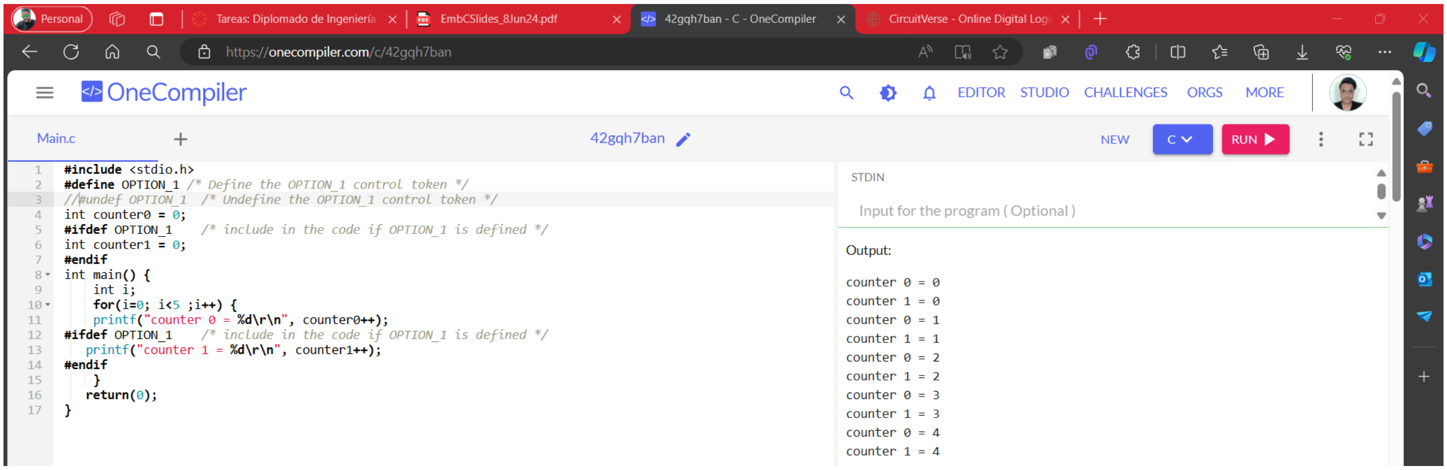
HandsOn: Conditional inclusion of code

Check the size of the data types for different architectures

- Review the following code to determine the expected output
- Capture and run the code to confirm
- Is it the expected behavior?
- After commenting out the third line of the code, is this the correct behavior?
- Modify the code to condition the inclusion of counter0 instead of counter1
- Report code and image of the results

```
#include <stdio.h>
#define OPTION_1 /* Define the OPTION_1 control token */
#undef OPTION_1  /* Undefine the OPTION_1 control token */
int counter0 = 0;
#ifdef OPTION_1   /* include in the code if OPTION_1 is defined */
int counter1 = 0;
#endif
int main() {
    int i;
    for(i=0; i<5 ;i++) {
    printf("counter 0 = %d\r\n", counter0++);
#ifdef OPTION_1   /* include in the code if OPTION_1 is defined */
   printf("counter 1 = %d\r\n", counter1++);
#endif
    }
   return(0);
}
```

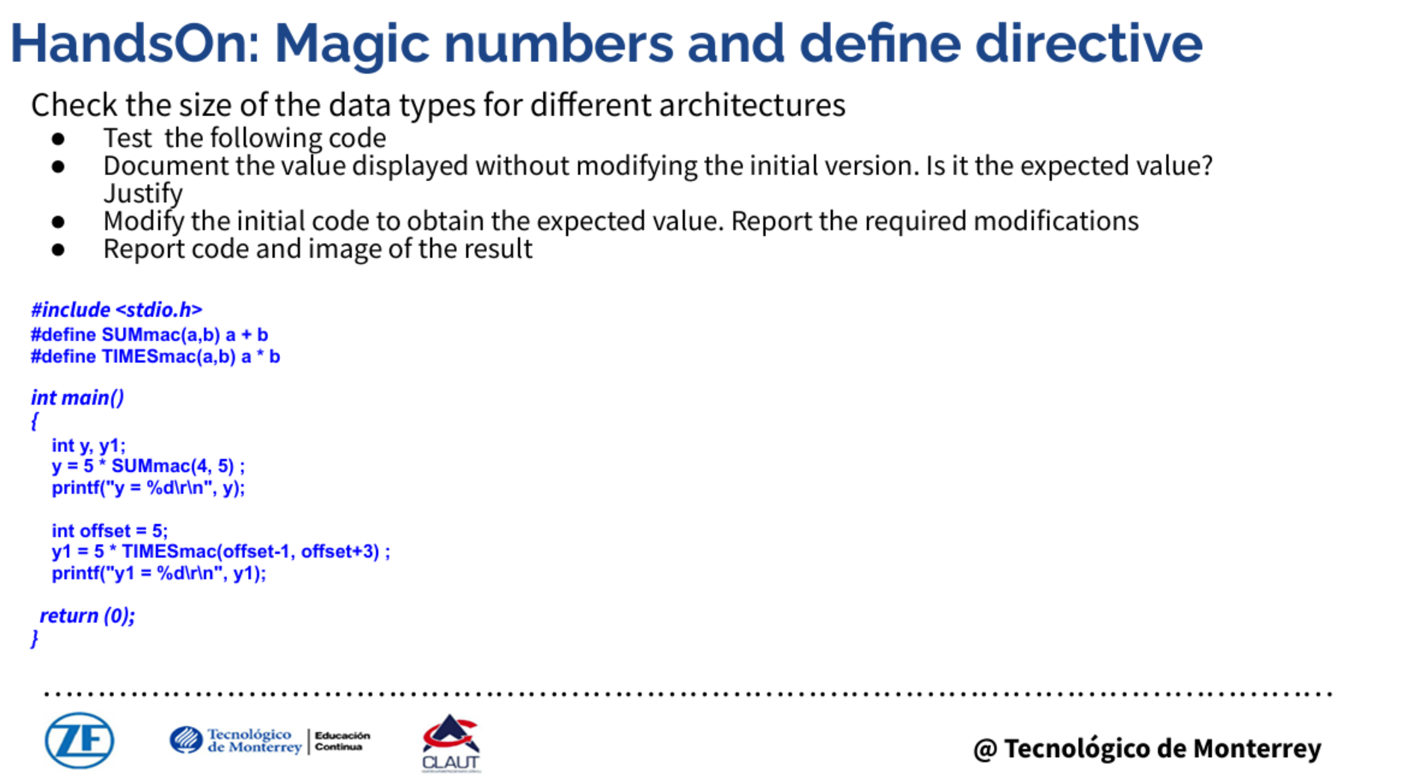
@ Tecnológico de Monterrey


Personal
Tareas: Diplomado de Ingeniería
EmbCSlides_8Jun24.pdf
42gqh7ban - C - OneCompiler
CircuitVerse - Online Digital Logi
https://onecompiler.com/c/42gqh7ban
OneCompiler
EDITOR
STUDIO
CHALLENGES
ORGS
MORE
Main.c
42gqh7ban
NEW
C
RUN
STDIN
Input for the program (Optional)

```
#include <stdio.h>
#define OPTION_1 /* Define the OPTION_1 control token */
//#undef OPTION_1  /* Undefine the OPTION_1 control token */
int counter0 = 0;
#ifdef OPTION_1    /* include in the code if OPTION_1 is defined */
int counter1 = 0;
#endif
int main() {
    int i;
    for(i=0; i<5 ;i++) {
    printf("counter 0 = %d\r\n", counter0++);
#ifdef OPTION_1    /* include in the code if OPTION_1 is defined */
   printf("counter 1 = %d\r\n", counter1++);
#endif
    }
   return(0);
}
```

Output:

```
counter 0 = 0
counter 1 = 0
counter 0 = 1
counter 1 = 1
counter 0 = 2
counter 1 = 2
counter 0 = 3
counter 1 = 3
counter 0 = 4
counter 1 = 4
```



HandsOn: Magic numbers and define directive

Check the size of the data types for different architectures

- Test the following code
- Document the value displayed without modifying the initial version. Is it the expected value? Justify
- Modify the initial code to obtain the expected value. Report the required modifications
- Report code and image of the result

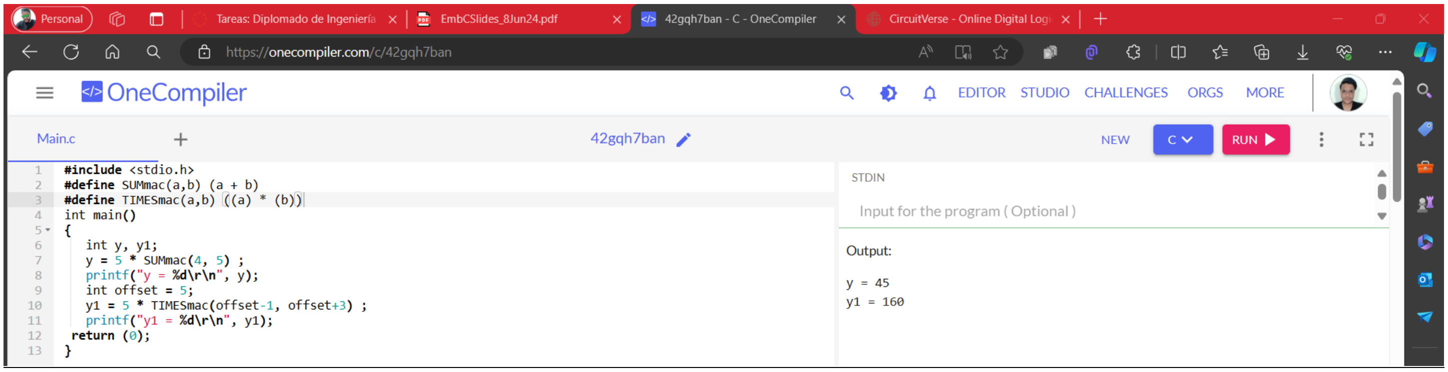
```
#include <stdio.h>
#define SUMmac(a,b) a + b
#define TIMESmac(a,b) a * b

int main()
{
   int y, y1;
   y = 5 * SUMmac(4, 5) ;
   printf("y = %d\r\n", y);

   int offset = 5;
   y1 = 5 * TIMESmac(offset-1, offset+3) ;
   printf("y1 = %d\r\n", y1);

 return (0);
}
```

@ Tecnológico de Monterrey



Personal
Tareas: Diplomado de Ingeniería
EmbCSlides_8Jun24.pdf
42gqh7ban - C - OneCompiler
CircuitVerse - Online Digital Logi
https://onecompiler.com/c/42gqh7ban
OneCompiler
EDITOR
STUDIO
CHALLENGES
ORGS
MORE
Main.c
42gqh7ban
NEW
C
RUN
STDIN
Input for the program (Optional)

```
#include <stdio.h>
#define SUMmac(a,b) (a + b)
#define TIMESmac(a,b) ((a) * (b))
int main()
{
   int y, y1;
   y = 5 * SUMmac(4, 5) ;
   printf("y = %d\r\n", y);
   int offset = 5;
   y1 = 5 * TIMESmac(offset-1, offset+3) ;
   printf("y1 = %d\r\n", y1);
 return (0);
}
```

Output:

```
y = 45
y1 = 160
```

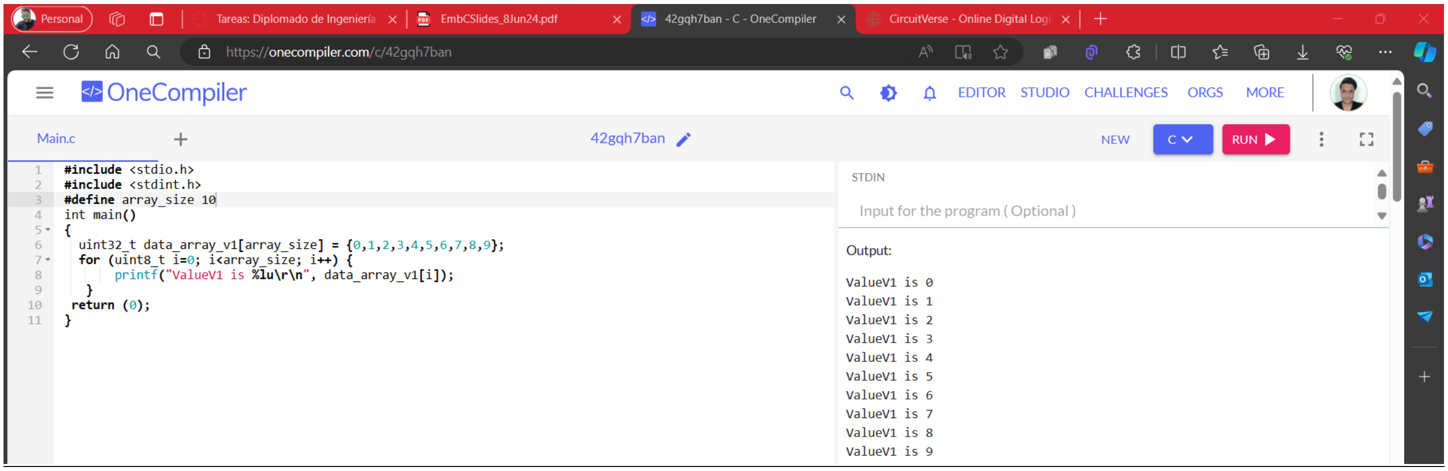
HandsOn: Magic numbers and define directive

Check the size of the data types for different architectures

- Test the following code
- Write a second version of the same code, but this time remove the magic number
- Write a third version of the same code, but this time increase the array size and values to 10
- Report code and image of the result

```
#include <stdio.h>
int main()
{
  uint32_t data_array_v1[5] = {0,1,2,3,4};

  for (i=0; i<5; i++) {
     printf("ValueV1 is %lu\r\n", data_array_v1[i]);
   }
 return (0);
}
```


Personal
Tareas: Diplomado de Ingeniería
EmbCSlides_8Jun24.pdf
42gqh7ban - C - OneCompiler
CircuitVerse - Online Digital Logi
https://onecompiler.com/c/42gqh7ban
OneCompiler
EDITOR
STUDIO
CHALLENGES
ORGS
MORE
Main.c
42gqh7ban
NEW
C
RUN
STDIN
Input for the program (Optional)

```
#include <stdio.h>
#include <stdint.h>
#define array_size 10
int main()
{
  uint32_t data_array_v1[array_size] = {0,1,2,3,4,5,6,7,8,9};
  for (uint8_t i=0; i<array_size; i++) {
       printf("ValueV1 is %lu\r\n", data_array_v1[i]);
   }
 return (0);
}
```

Output:

```
ValueV1 is 0
ValueV1 is 1
ValueV1 is 2
ValueV1 is 3
ValueV1 is 4
ValueV1 is 5
ValueV1 is 6
ValueV1 is 7
ValueV1 is 8
ValueV1 is 9
```

HandsOn: Use of suffix

Check the size of the data types for different architectures

- Capture and run the code at the web tool. What is the problem? Explain
- Incorporate the proper corrections and run the code again. Was the problem solved? Record the modifications needed at the original code.

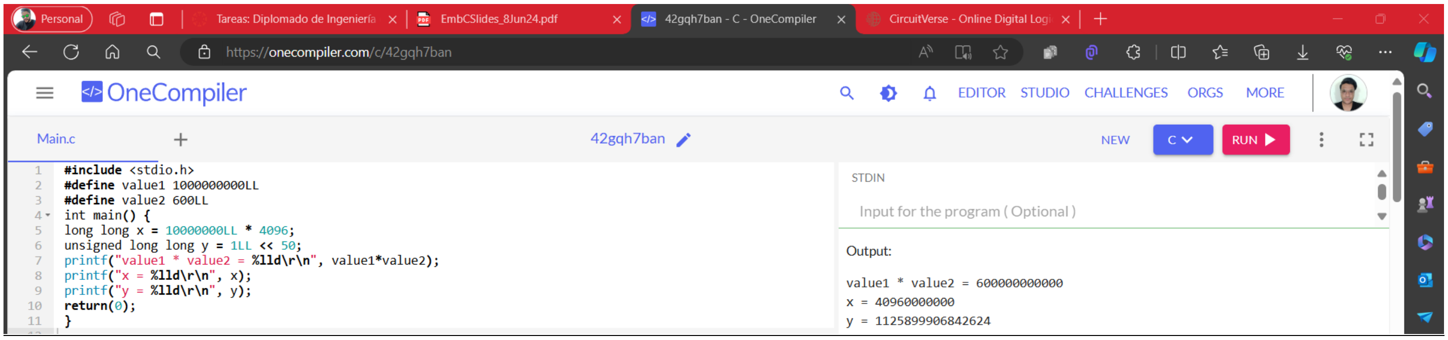
```
#include <stdio.h>
#define value1 1000000000LL
#define value2 600


int main() {

        long long x = 10000000 * 4096;
        unsigned long long y = 1 << 50;

        printf("value1 * value2 = %lld\r\n", value1*value2);
        printf("x = %lld\r\n", x);
        printf("y = %lld\r\n", y);

        return(0);
}
```


Personal
Tareas: Diplomado de Ingeniería
EmbCSlides_8Jun24.pdf
42gqh7ban - C - OneCompiler
CircuitVerse - Online Digital Logi
https://onecompiler.com/c/42gqh7ban
OneCompiler
EDITOR
STUDIO
CHALLENGES
ORGS
MORE
Main.c
42gqh7ban
NEW
C
RUN
STDIN
Input for the program (Optional)

```
#include <stdio.h>
#define value1 1000000000LL
#define value2 600LL
int main() {
long long x = 10000000LL * 4096;
unsigned long long y = 1LL << 50;
printf("value1 * value2 = %lld\r\n", value1*value2);
printf("x = %lld\r\n", x);
printf("y = %lld\r\n", y);
return(0);
}
```

Output:

```
value1 * value2 = 600000000000
x = 40960000000
y = 1125899906842624
```