# Session 3

Mostafa Akram

User

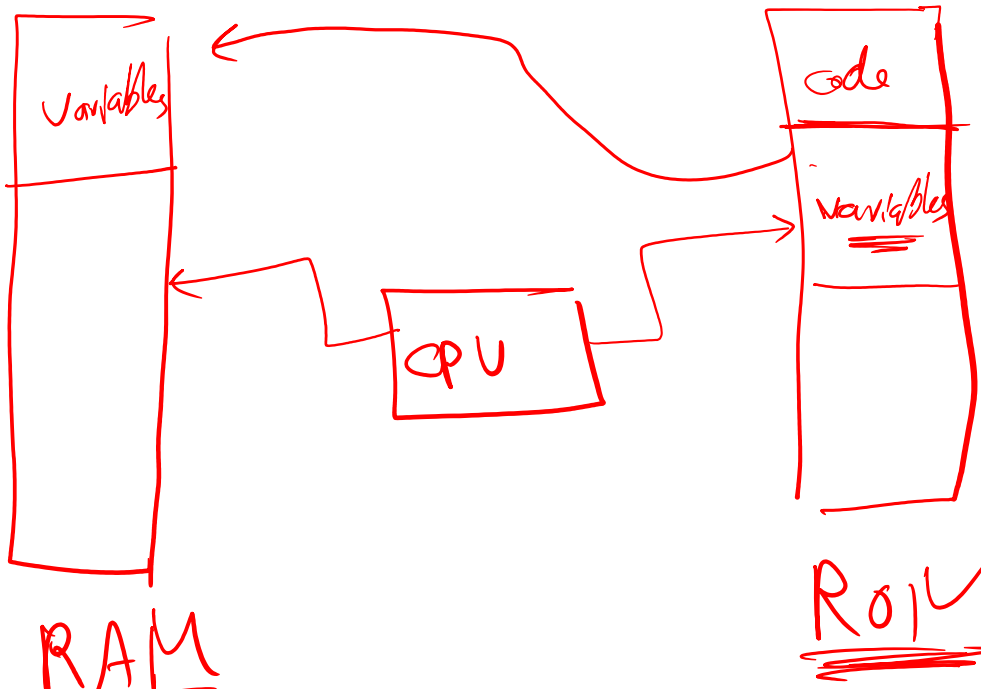
Page → register Page

register New User
name → string
birth date → date
gender -
mail → @ -.com
phone number → '0' + number

→ Programmer → string name;
→ string birthdate;
→ int phonenumber
.
.
.

device

code size 300 kb
256 Kb




Embedded Target - → Micro controller

# Variables

| Data Type | Range | Bytes | Format |
|---|---|---|---|
| signed char | -128 to + 127 | 1 | %c |
| unsigned char | 0 to 255 | 1 | %c |
| short signed int | -32768 to +32767 | 2 | %d |
| short unsigned int | 0 to 65535 | 2 | %u |
| signed int | -32768 to +32767 | 2 | %d |
| unsigned int | 0 to 65535 | 2 | %u |
| long signed int | -2147483648 to +2147483647 | 4 | %ld |
| long unsigned int | 0 to 4294967295 | 4 | %lu |
| float | -3.4e38 to +3.4e38 | 4 | %f |
| double | -1.7e308 to +1.7e308 | 8 | %lf |
| long double | -1.7e4932 to +1.7e4932 | 10 | %Lf |

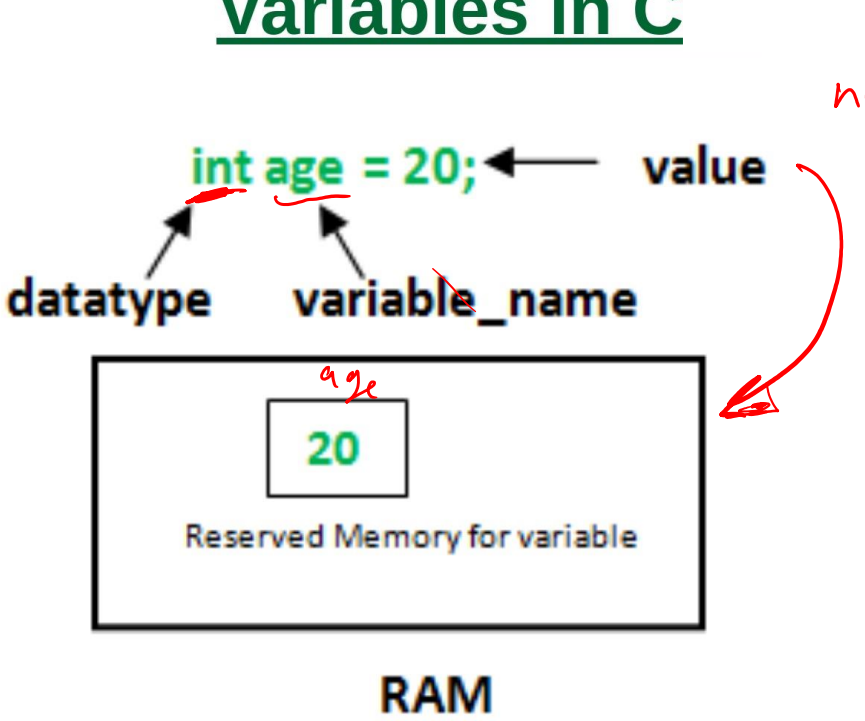
Note: The sizes and ranges of int, short and long are compiler dependent. Sizes in this figure are for 16-bit compiler.

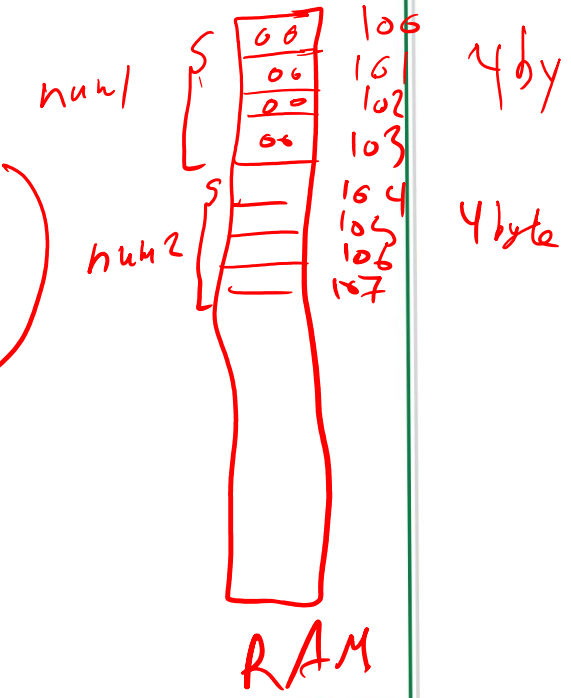
for "sizeof" operator → %zu

for address operator → %p


# Variables

## Variables in C



# Variables

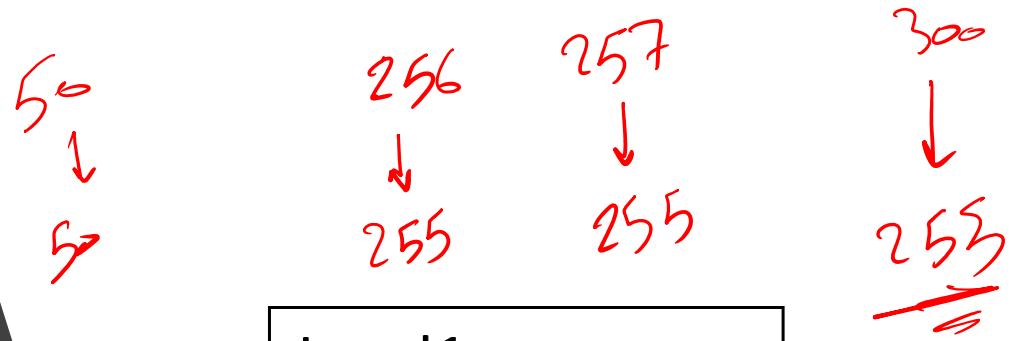
## Variable Naming (Identifiers)

- Variable name includes:
  - Letters a…z, A…Z
  - Numbers 0…9
  - Underscore _
- First character either letters (a…z, A…Z) or underscore ( _ )
- Variable name doesn't include:
  - Spaces
  - Other characters ! , @ : # …
  - Reserved names (int, float, void … etc)
- A ≠ a

# Ascii code

```
cook@pop-os:~$ ascii -d
   0 NUL    16 DLE    32      48 0    64 @    80 P    96 `   112 p
   1 SOH    17 DC1    33 !    49 1    65 A    81 Q    97 a   113 q
   2 STX    18 DC2    34 "    50 2    66 B    82 R    98 b   114 r
   3 ETX    19 DC3    35 #    51 3    67 C    83 S    99 c   115 s
   4 EOT    20 DC4    36 $    52 4    68 D    84 T   100 d   116 t
   5 ENQ    21 NAK    37 %    53 5    69 E    85 U   101 e   117 u
   6 ACK    22 SYN    38 &    54 6    70 F    86 V   102 f   118 v
   7 BEL    23 ETB    39 '    55 7    71 G    87 W   103 g   119 w
   8 BS     24 CAN    40 (    56 8    72 H    88 X   104 h   120 x
   9 HT     25 EM     41 )    57 9    73 I    89 Y   105 i   121 y
  10 LF     26 SUB    42 *    58 :    74 J    90 Z   106 j   122 z
  11 VT     27 ESC    43 +    59 ;    75 K    91 [   107 k   123 {
  12 FF     28 FS     44 ,    60 <    76 L    92 \   108 l   124 |
  13 CR     29 GS     45 -    61 =    77 M    93 ]   109 m   125 }
  14 SO     30 RS     46 .    62 >    78 N    94 ^   110 n   126 ~
  15 SI     31 US     47 /    63 ?    79 O    95 _   111 o   127 DEL
```

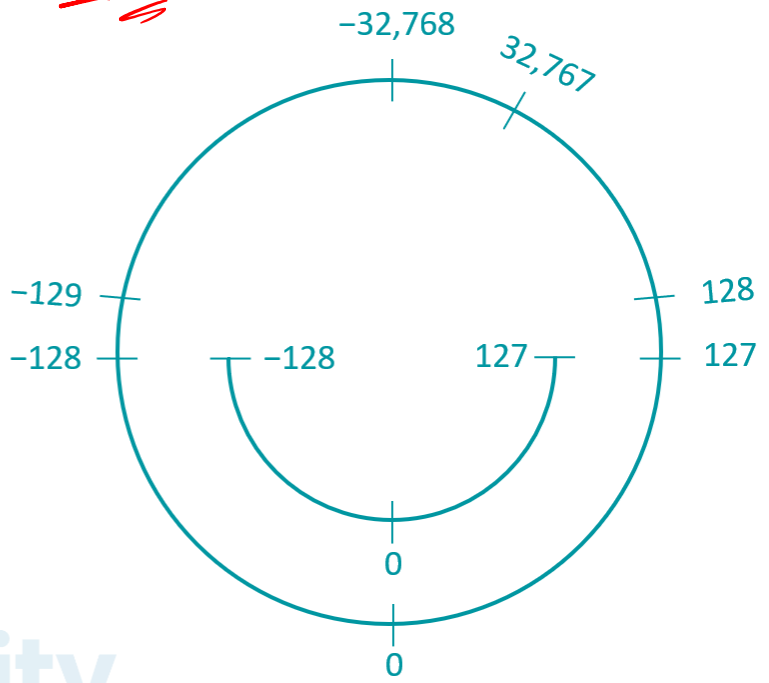
# Variable overflow




**Legend 1**

Inner arc: signed `char`

Outer circle: signed `short int`




# Variables

| Operators | Meanings |
|---|---|
| + | Addition or unary plus |
| - | Subtraction or unary minus |
| * | Multiplication |
| / | Division |
| % | Modulo division |

**Table 1: Arithmetic Operators in C**

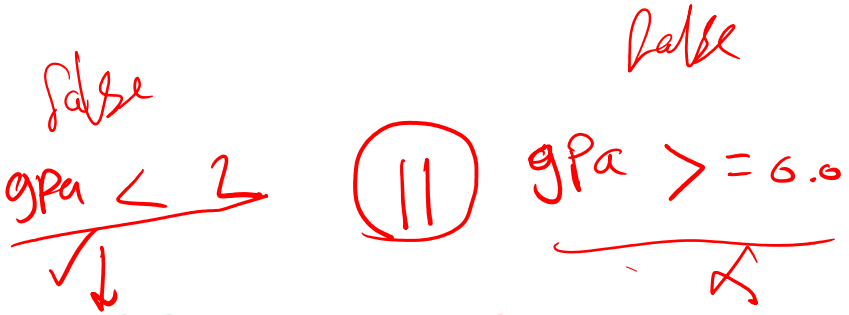
4%2 = 0 → 4/2 = 2

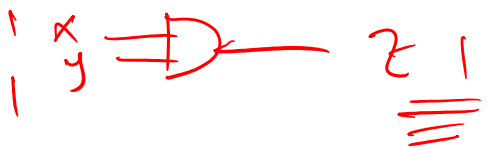
5%2 = 1 → 5/2 = 2 → 1

17%3 = 2 → 15/3 = 5 → 2

# If .. Else statement

```
if (condition)
{
    // execute code statements of if block when
    // condition is true
}
 else
{
    // execute code statements of if block when
    // condition is true
}
```



Switch statement

```
switch(variable)
{
    case 1:
            //execute your code
    break;
    case n:
            //execute your code
    break;
    default:
            //execute your code
    break;
}
```

# Switch statement

## Rules of the switch case statement

Following are some of the rules that we need to follow while using the switch statement:

1. In a switch statement, the "**case value**" must be of "**char**" and "**int**" type.
2. There can be one or N number of cases.
3. The values in the case must be **unique**.
4. Each statement of the case can have a break statement. It is optional.
5. The default Statement is also optional.

# Task

- The hostel in which you plan to spend the night tonight offers very interesting rates, as long as you do not arrive too late. Housekeeping finishes preparing rooms by noon, and the sooner guests arrive after noon, the less they have to pay. You are trying to build a C program that calculates your price to pay based on your arrival time.
- Your program will read an integer (between 0 and 12) indicating the number of hours past noon of your arrival. For example, 0 indicates a noon arrival, 1 a 1pm arrival, 12 a midnight arrival, etc. The base price is 10 dollars, and 5 dollars are added for every hour after noon. Thankfully the total is capped at 53 dollars, so you'll never have to pay more than that. Your program should print the price (an integer) you have to pay, given the input arrival time.

### Example 1

Input

```
7
```

Output

```
45
```

### Example 2

Input

```
10
```

Output

```
53
```

# Task

- You arrive in front of a bridge that you must cross to reach a village before dark. Crossing the bridge is not free - the bridge keeper asks you to roll two dice to determine the cost. You decide to write a program to verify that he is charging the right price.
- Your program should read two integers, between 1 and 6, representing the values of each die. If the sum is greater than or equal to 10, then you must pay a special fee (36 coins). Otherwise, you pay twice the sum of the values of the two dice. Your program must then display the text "Special tax" or "Regular tax" followed by the amount you have to pay on the next line.

Example

Input

```
5
6
```

Output

```
Special tax
36
```

Input

```
4
3
```

Output

```
Regular tax
14
```

# Links