Variables, Data types

CHAPTER 13

SURESH TECHS

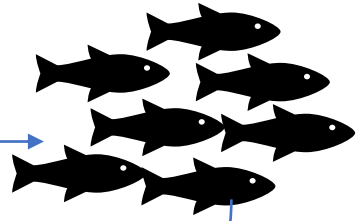
C PROGRAMMING COURSE

NOTE

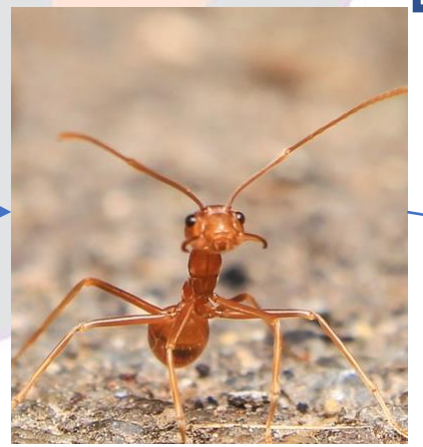
ఈ చాప్టర్ లో కుంచుం ఎక్కువ discuss చేస్తాను. కాబట్టి, మొదటి సారి అన్నీ అర్థం అయ్యే అవకాశం లేదు.

ఒకవేళ అర్థం కాకపోయినా పర్వాలేదు, next lessons చూస్తున్నపుడు automatic గా అర్థం అవుతాయి. కానీ, ఈ chapter లో చెప్పిన విషయాలు గుర్తుంటే మాత్రం మీరు ఎటువంటి programming ఐనా కూడా easy గా నేర్చుకోగలరు.

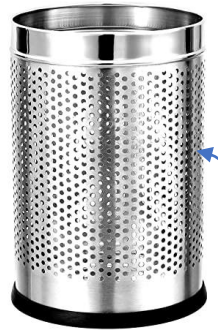
7 చేపల కథ



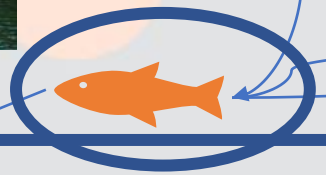
నా బంగారు పుట్టలో
వేలు పెడితే కుట్టనా



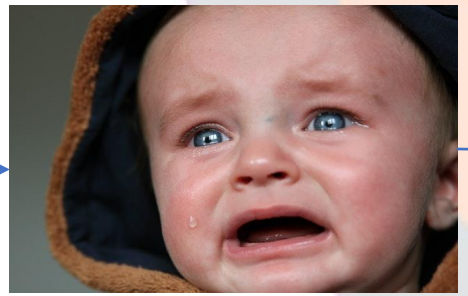
7 చేపల కథ

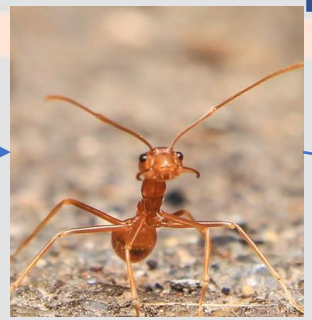


Sometimes we will stop!



నా బంగారు పుట్టలో
వేలు పెడితే కుట్టనా





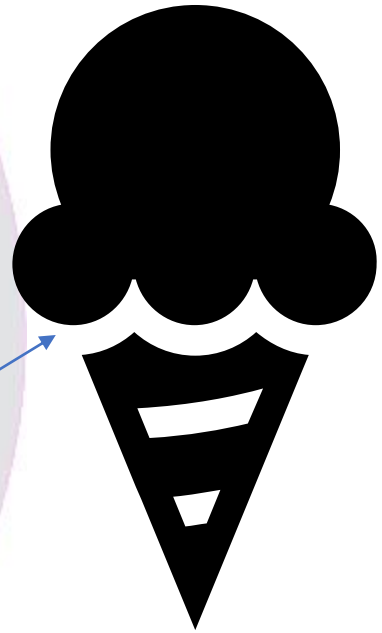
Sometimes we will discuss deep concepts

Data types, Variables

Childhood days (చిన్ననాటి రోజులు)

pot-a

pot-b



pot-a = 10

pot-b = 30

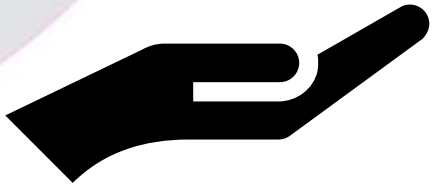
Variable

```c
#include<stdio.h>
int main(){
    int a;
    int b;
    a = 10;
    b = 30;
    int sum = a+b;
    printf("%d\n",sum);
    a = 90;
    int sub = a-30;
    b = sub;
    printf("a = %d, b = %d",a,b);
    return 0;
}
```

- It is **used to store data**
- It's **value can be changed at any time**

a

b



కొన్ని రోజులు (Research)

కొన్ని రోజుల తరువాత

Dennis Ritchie

1. 10 bores are not working – int bores = 10;

2. Pass percentage of school students are 30.26

3. Fight with king “SURBALI”

Let’s operate our kingdom

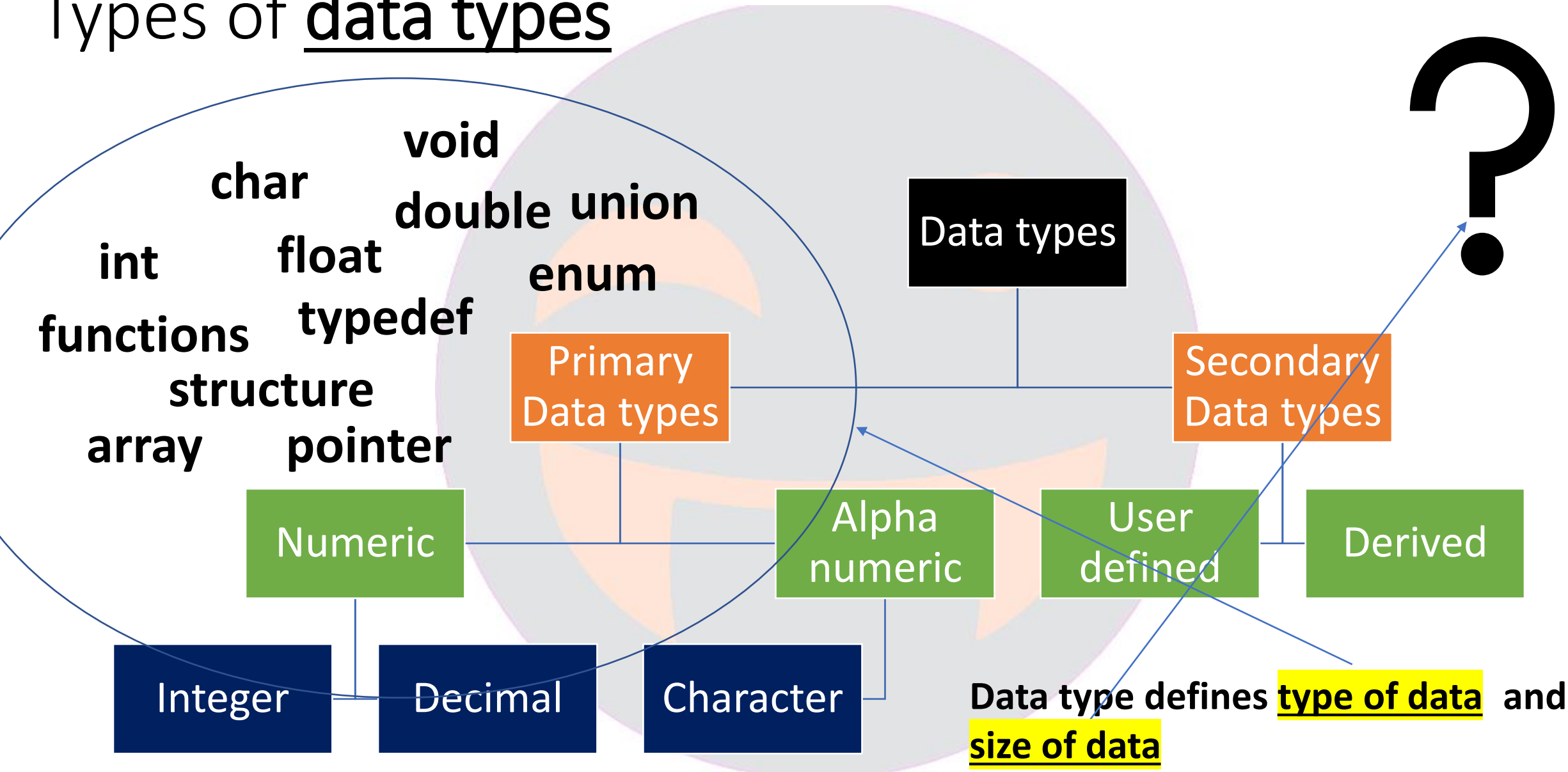
Father

నాన్న నాకు కుంచం సమయం ఇవ్వండి

SURBALI

- A **program** can have **different types** of data
  - 10 int
  - 30.26 float, double
  - SURBAN char
- Discovered **types for data** and named them as **data types**

Types of **data types**



Data type defines type of data and size of data

Small Math

- 500
- 5*100
- 5*10^2
- 5e2

- 80000
- 8*10^4
- 0.8*10^5
- 0.8e5

Mass/Weight of the Earth?

5970,000,000,000,000,000,000,000 kg

597*10^22

5.97*10^24

5.97e24

mantissa e exponent

Real number expressed in decimal notation or an integer

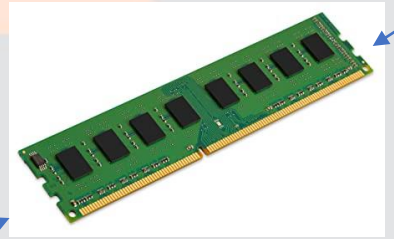
Integer number with an optional plus or minus sign

Letter e separating mantissa and exponent can be written lowercase or uppercase

What is memory?

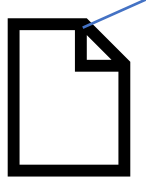
Registers usually consist of a small amount(32 bit, 64 bit) of fast storage

Processor(cup) – i3, i5, i7



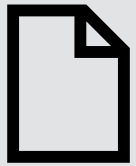
RAM (4gb, 8gb, 16gb, 32gb)

ROM (4mb, 8mb)



printfprograms.exe
/printfprograms.out

Compiler



printfprograms.c

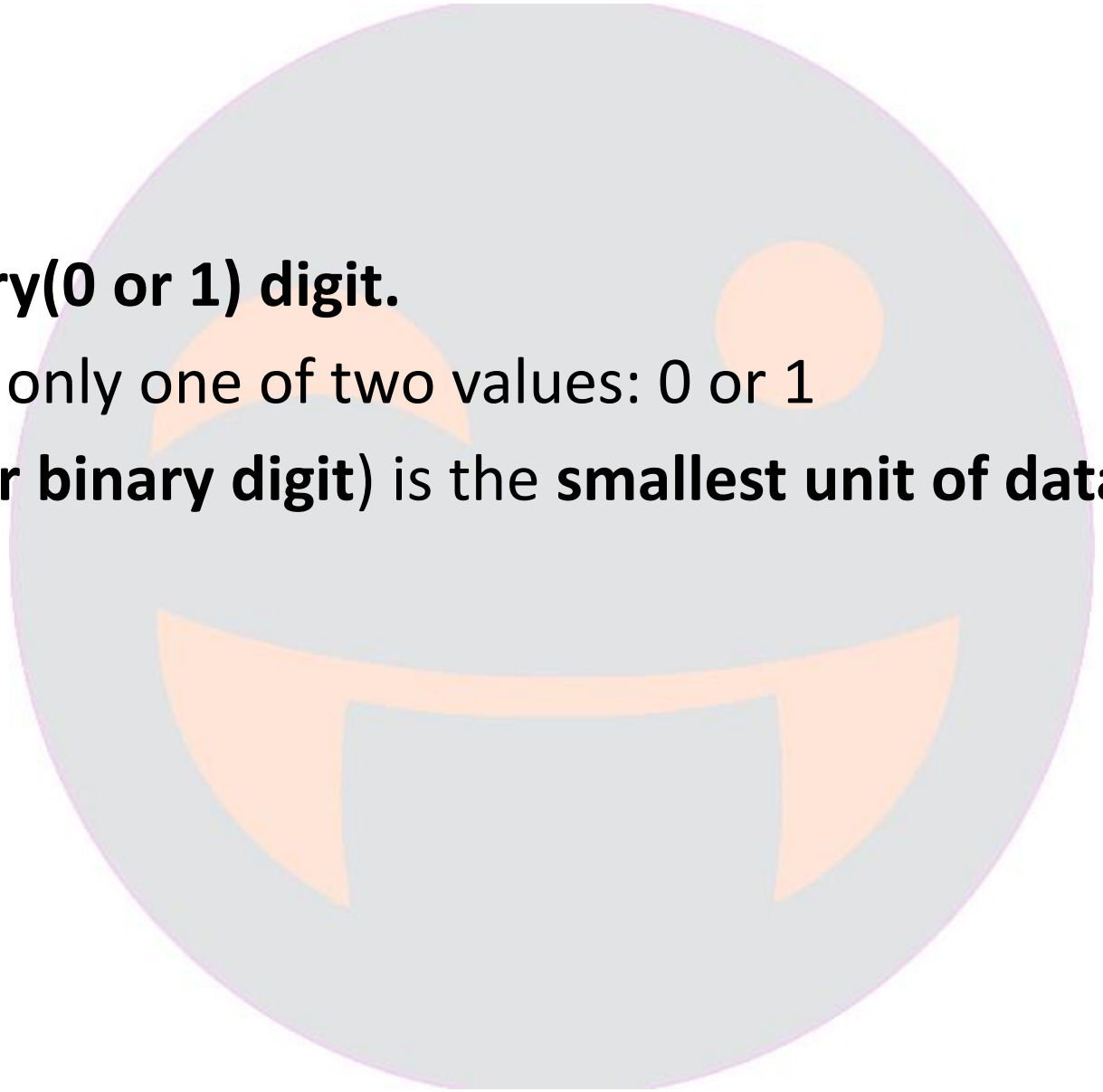
D:\CPrograms\Excersie1

HARD DISK (500gb,1tb,2tb)

Small test

- You have **S, U characters** and you have to come up with different combinations with **size(length) of 2**
- SU
- US
- UU
- SS

- A bit is a **binary(0 or 1) digit.**
- A bit can hold only one of two values: 0 or 1
- A bit (**short for binary digit**) is the **smallest unit of data in a computer**



- 00
- 01
- 10
- 11

Combinations(0 & 1 – size of **2 bits**)

- 000
- 001
- 010
- 100
- 101
- 110
- 011
- 111

2^3 = 8

- 00
- 01
- 10
- 11

2^2 = 4

Combinations(0 & 1 – size of **3 bits**)

8 bits(01001101) - 1 byte

2^8 = 2*2*2*2*2*2*2*2 = 256

With 1 byte, we can represent 256 numbers

data type defines size of the data

More the memory, more the size 😍 😍

- 1 byte – 8 bits
- 2 bytes – 16 bits
- 4 bytes – 32 bits

Memory

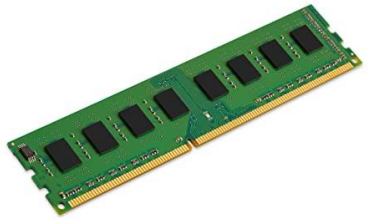
int a = 10;

- **2^8 = 256**
- **2^16 = 65,536**
- **2^32= 4,29,49,67,296**

Small - Rs. 10

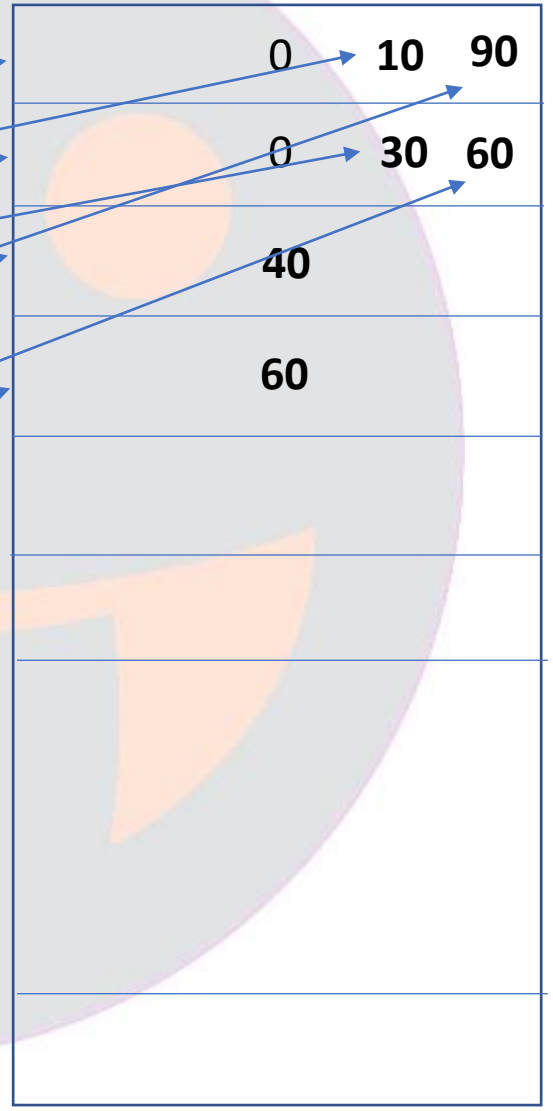
Medium – Rs. 20

Large - Rs. 40



```c
#include<stdio.h>
int main(){
    int a;
    int b;
    a = 10;
    b = 30;
    int sum = a+b;
    printf("%d\n",sum);
    a = 90;
    int sub = a-30;
    b = sub;
    printf("a = %d, b = %d",a,b);
    return 0;
}
```

```
40
a = 90, b = 60
```



0011010001101011 0

0011010001101011 1

0011010001101010 1

0011010001101000 1

What is data type?

- **Data type defines type of data and size of data**
- **Size is nothing but memory – how much data that variable can hold**

Basic data types(most used)

Type	Size(bytes)	Range	Format specifier
int	2 / **4**	-2,14,74,83,648 to 2,14,74,83,647	%d, %i
char	1	-128 to 127	%c
float	4	3.4E +/-38	%f
double	8	1.7E +/-308	%lf

https://opensource.apple.com/source/xnu/xnu-792.13.8/EXTERNAL_HEADERS/ppc/limits.h.auto.html

https://opensource.apple.com/source/Libm/Libm-47.1/i386.subproj/float.h.auto.html

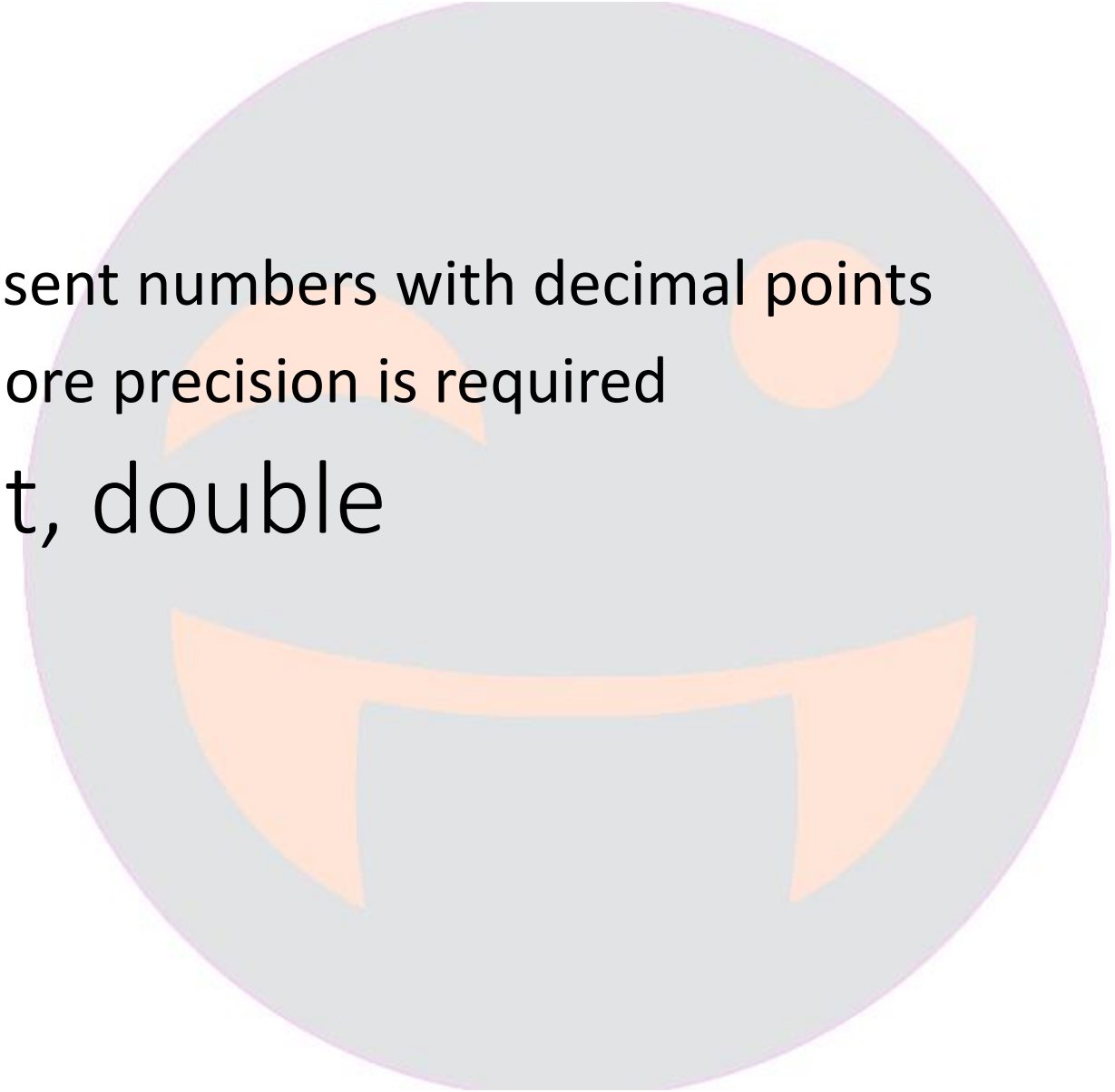
- Used to represent whole numbers that can have zero, positive and negative values but no decimal values
- We use **int** for declaring a integer variable

int

- **int a;**
- We can declare multiple variables at once
- int a,b,c;

- Used to represent numbers with decimal points
- Used when more precision is required

float, double

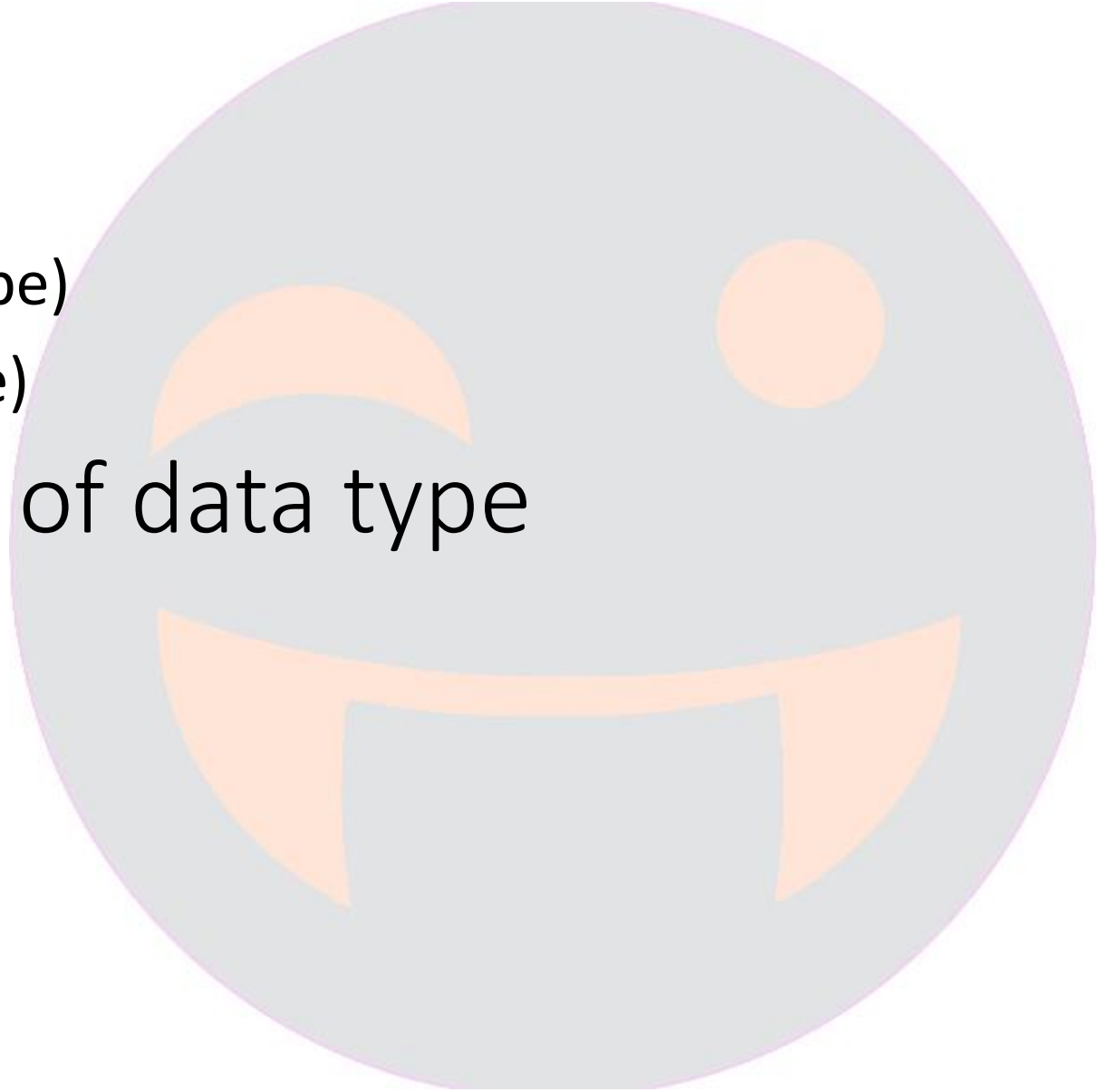


Difference between Float and Double

Float	Double
4 bytes – 32 bits	8 bytes – 64 bits
Used to represent decimal values(ex: 7.987)	Used to represent decimal values(ex: 8.182)
Single precision data type(6 decimal places) Ex: 7.879888	Double precision data type(15 decimal places) Ex: 8.182191283871221
It is faster	It is slow as it works with very large values

- sizeof(data type)
- sizeof(variable)

Size of data type



```c
#include<stdio.h>
int main(){
    int noOfBores = 10;
    float passPercentage = 30.26;
    printf("%d\n",noOfBores);
    printf("%f",passPercentage);
    return 0;
}
```

- int noOfBores = 10;
- float passPercentage = 30.26;
- What about SURBALI 🤣🤣?

SURBALI

How do we represent a character?

Collection of characters is called a string

char data type

- It is used to store **single character values**
- **C** uses a character encoding scheme called ASCII to represent characters or special characters
- ASCII – **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange

char data type

- Each **character has an ASCII value associated with it**.
- The keyword “**char**” is used to declare the char data type.
- The format specifier for this data type is "**%c**“

Storing characters and letters

- char data type is used to store characters/letters
- Underneath C stores integer numbers instead of characters
- Note: In order to represent characters, the computer has to map each integer with a corresponding character using a numerical code. The most common numerical code is ASCII, which stands for American Standard Code for Information Interchange

```
for(int i=-128;i<=127;i++){
    printf("%d - %c\t",i,i);
}
```

char values are stored in 1 byte in memory, and value range from -128 to 127 or 0 to 255

- Array?
- **In simple terms:** Collection of similar items is called an **array**

String – sequence of characters(SUBBALU)

humans[]

Strings

- **char** – to represent one character.
- yes – collection of y, e, s
- String is a **sequence of characters** terminated(end) with a null character **\0**
- **Ex:** char yes[]="yes"
- **Compiler appends null(NUL) character \0 at the end**

```c
#include<stdio.h>
int main(){
    char yes='y';
    char no='n';
    printf("Welcome to suresh techs\n");
    printf("Yes: %c",yes);
    return 0;
}
```

y	e	s	\0
0	1	2	3

Declaring string

- Two ways to declare a string in c
- String Literal
- Char array

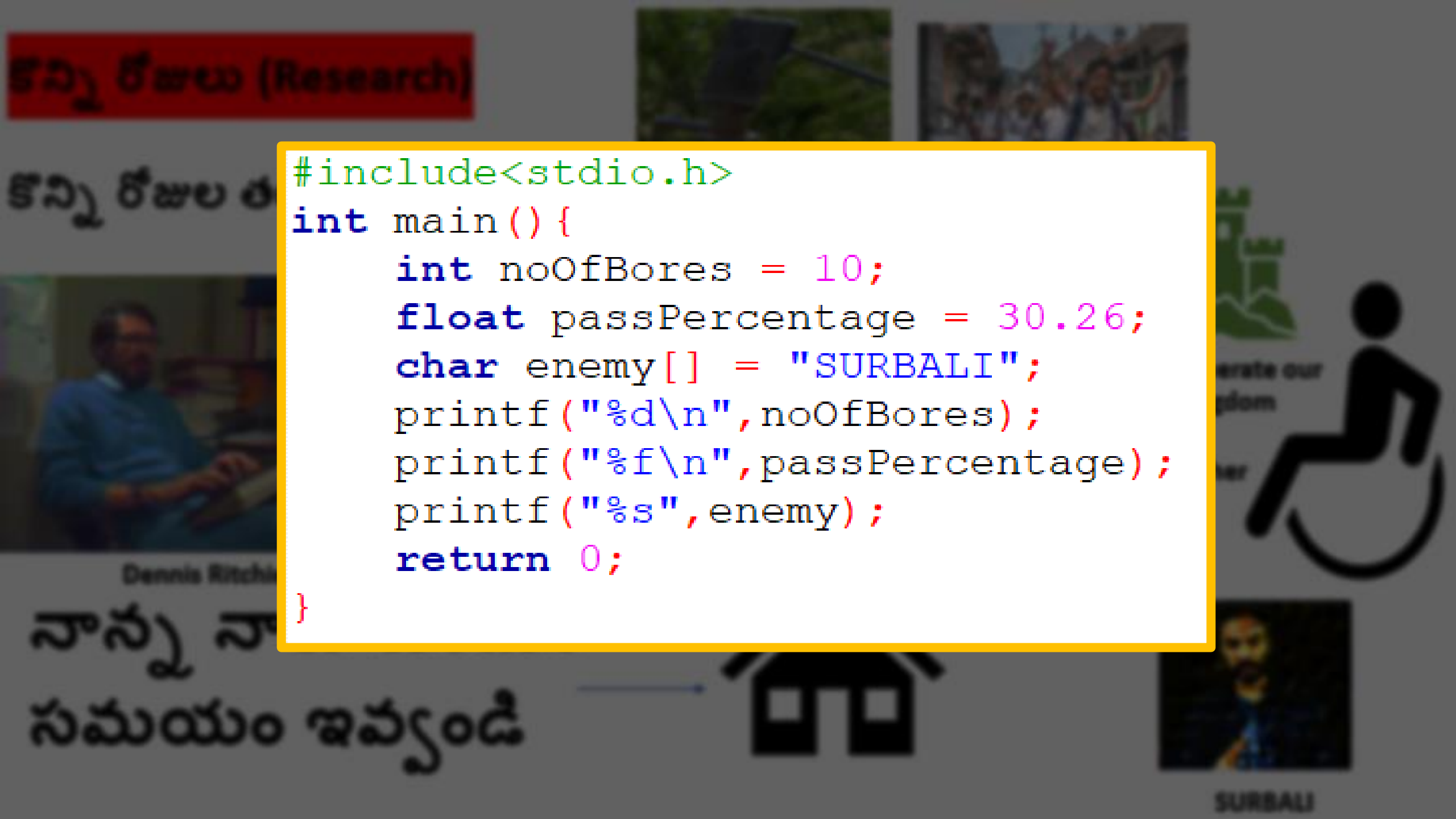
- char name[]="suresh"; //assigns memory automatically(string literal)
- char name[10]="suresh"; //we have assigned 10 bytes of memory

String literal

- char name[]={‘s’,’u’,’r’,’e’,’s’,’h’,’\0’}; //declaring using char array
- What happens if you put more than 10 bytes🤔🤔
- It may do anything it wants: It may scold you, it say “taggede le” or it can ask you to like this video. Literally anything…

Char array

NOTE: Will discuss more about strings later on




```c
#include<stdio.h>
int main(){
    int noOfBores = 10;
    float passPercentage = 30.26;
    char enemy[] = "SURBALI";
    printf("%d\n",noOfBores);
    printf("%f\n",passPercentage);
    printf("%s",enemy);
    return 0;
}
```

int a;

type = int
size = 2 or **4 bytes**
Range = -2,14,74,83,648 to 2,14,74,83,647

- **It is used to store data**
- **It's value can be changed at any time**
- Every variable in c has a specific **type,** which determines the **size of the memory** and the **range of values** that can be stored within that memory.

Variable - actu

```c
#include<stdio.h>
int main(){
    int a;
    int b;
    a = 10;
    b = 30;
    int sum = a+b;
    printf("%d\n",sum);
    a = 90;
    int sub = a-30;
    b = sub;
    printf("a = %d, b = %d",a,b);
    return 0;
}
```

```
40
a = 90, b = 60
```

Variable

- It is used to store data
- It's value can be changed at any time

a

b

Declaration of variables

User defined type declaration – will be discussed later

- **Primary type declaration**
  - data-type v;
  - data-type v1,v2,….vn;
- Ex:
  - int sum;
  - int total, allCount;
  - float percentage;

Further classification – From basic data types

Basic data types:

int, char, float, double

After introducing modifiers:

- int
  - signed int
  - unsigned int
  - long int
  - short int

Modifiers:

signed **- To represent signed values(+ and -)**

unsigned **- To represent only positive values(+)**

long

short **- affects the range of values(long, short)**

Signed and unsigned – int

- int –(32 bit environment) - 4 bytes – 32 bits
- Range: 2^32 => 4,29,49,67,296
- Signed: (positive and negative) => -2,147,483,647 to 2,147,483,647
- Unsigned: (only positive)=> 0 to 4,29,49,67,296

Default is signed

All basic data types with modifiers

Type	Typical Size in Bits	Minimal Range	Format Specifier
`char`	8	-127 to 127	`%c`
`unsigned char`	8	0 to 255	`%c`
`signed char`	8	-127 to 127	`%c`
`int`	16 or 32	-32,767 to 32,767	`%d`, `%i`
`unsigned int`	16 or 32	0 to 65,535	`%u`
`signed int`	16 or 32	Same as int	`%d`, `%i`
`short int`	16	-32,767 to 32,767	`%hd`
`unsigned short int`	16	0 to 65,535	`%hu`
`signed short int`	16	Same as short int	`%hd`
`long int`	32	-2,147,483,647 to 2,147,483,647	`%ld`, `%li`
`long long int`	64	$-(2^{63} - 1)$ to $2^{63} - 1$ (Added by C99 standard)	`%lld`, `%lli`
`signed long int`	32	Same as long int	`%ld`, `%li`
`unsigned long int`	32	0 to 4,294,967,295	`%lu`
`unsigned long long int`	64	$2^{64} - 1$ (Added by C99 standard)	`%llu`
`float`	32	1E-37 to 1E+37 with six digits of precision	`%f`
`double`	64	1E-37 to 1E+37 with ten digits of precision	`%lf`
`long double`	80	1E-37 to 1E+37 with ten digits of precision	`%Lf`

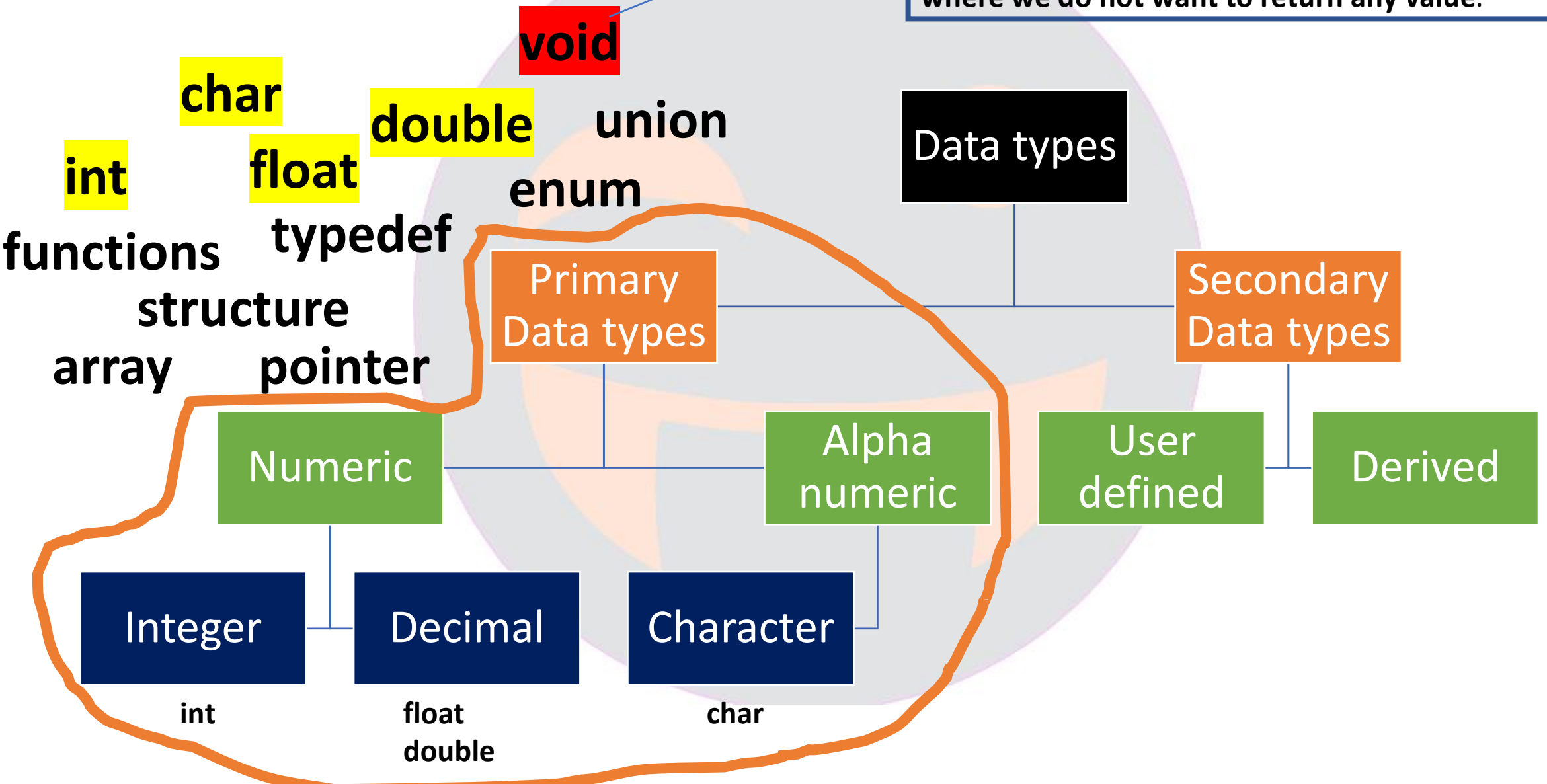
19 decimal places

- You have to store **half glass of water** in one of the containers and you **can’t use the container for another purpose**

Choose the best one

Primary data types

void: Special type known as **empty data type** that is used to state that a given variable does not have any type. This **is mainly used in defining functions where we do not want to return any value**.



void is not used for declaring variables

```c
#include<stdio.h>
int main(){
    int noOfBores = 10;
    float passPercentage = 30.26;
    char enemy[] = "SURBALI";
    printf("%d\n",noOfBores);
    printf("%f\n",passPercentage);
    printf("%s",enemy);
    return 0;
}
```

- void noOfBores = 10;
- void marks = 200;
- Above are invalid declarations

void is used with functions

```c
#include<stdio.h>
int sumByFifty(int n){
    return n+50;
}

int main(){
    printf("Welcome\n");
    int sum = sumByFifty(30);
    int result = sum*2;
    printf("%d",result);
    return 0;
}
```

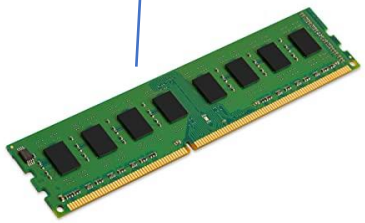
```c
#include<stdio.h>
void sumByFifty(int n){
    int sum = n+50;
    printf("%d",sum);
}

int main(){
    printf("Welcome\n");
    sumByFifty(30);
    return 0;
}
```

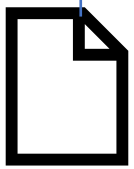
```c
#include<stdio.h>
void sumByFifty(void){
    int sum = 50;
    printf("%d",sum);
}

int main(){
    printf("Welcome\n");
    sumByFifty();
    return 0;
}
```

It is used **if nothing is passed in the function** or the **function doesn't return anything**.

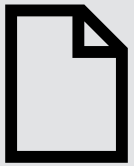


RAM (4gb, 8gb, 16gb, 32gb)



printfprograms.exe /printfprograms.out

Compiler



printfprograms.c

D:\CPrograms\Excersie1

HARD DISK (500gb,1tb,2tb)

int a;

type = int
size = 2 or **4 bytes**
Range = -2,14,74,83,648 to 2,14,74,83,647

Rules to define variables

- It is used to **store data**
- It's value can be **changed at any time**
- Every variable is associated with a specific **type,** which determines the **size of the memory** and the **range of values** that can be stored within that memory.

```
int a;
int b;
a = 10;
b = 30;
int sum = a+b;
printf("%d\n",sum);
a = 90;
int sub = a-30;
b = sub;
printf("a = %d, b = %d",a,b);
return 0;
}
```

```
40
a = 90, b = 60
```

- It is used to store data
- It's value can be changed at any time

a

b

Case sensitive

- Who is the father of C Language?
- Dennis Ritchie
- dennis Ritchie
- Dennis ritchie

- C language is case sensitive

```c
#include<stdio.h>
int main(){
    int a;
    int b;
    a = 10;
    b = 30;
    int sum = a+b;
    printf("%d\n",Sum);
    printf("%d, %d\n",a,b);
    a = 90;
    int sub = a-b;
    printf("%d",sub);
}
```

1. Variable names may consists of **letters, digits, and the underscore(_) characters**

Rules to define variables

```c
#include<stdio.h>
int main(){
    int a;
    int b;
    a = 10;
    b = 30;
    int sum = a+b;
    printf("%d\n",sum);
}
```

```c
#include<stdio.h>
int main(){
    int moneypot1;
    int b;
    moneypot1 = 10;
    b = 30;
    int sum = moneypot1+b;
    printf("%d\n",sum);
}
```

2. They **must begin with either letter or underscore**

Ru

```c
#include<stdio.h>
int main(){
    int _moneypot1;
    int 2moneypot;
    a = 10;
    b = 30;
    int sum = a+b;
    printf("%d\n",sum);
}
```

3. **White space is not allowed, instead use _ or camelCase**

- Ex: total_sum, totalSum

```
#include<stdio.h>
int main(){
    int moneypot1;
    int moneypot2;
    moneypot1 = 10;
    moneypot2 = 30;
    int total sum = moneypot1+moneypot2;
    printf("%d\n",total sum);
}
```

4. Variable names **case sensitive** – Uppercase and lowercase are significant.

Rules to define variables

```c
#include<stdio.h>
int main(){
    int moneypot1;
    int moneypot2;
    moneypot1 = 10;
    moneypot2 = 30;
    int totalSum = moneypot1+moneypot2;
    printf("%d\n",totalsum);
}
```

5. Variable name **should not be a keyword**

```c
#include<stdio.h>
int main(){
    int char = 10;
    int moneypot2;
    char = 10;
    moneypot2 = 30;
    int totalSum = char+moneypot2;
    printf("%d\n",totalSum);
}
```

- ANSI standard recognizes a length of 31 characters for a variable name. However, the length should **not** be normally **more than** any combination of **eight alphabets, digits, and underscores**.

Note – happens only in few

```
#include<stdio.h>
int main(){
    int moneypotInMyFirstAlmaraPlacedOnTopOftheFirstFloor;
    int moneypotInMyFirstAlmaraPlacedOnTopOftheFirstFloor1;

    moneypotInMyFirstAlmaraPlacedOnTopOftheFirstFloor = 10;
    moneypotInMyFirstAlmaraPlacedOnTopOftheFirstFloor1 = 30;

    int totalSum =
moneypotInMyFirstAlmaraPlacedOnTopOftheFirstFloor+
moneypotInMyFirstAlmaraPlacedOnTopOftheFirstFloor1;

    printf("%d\n",totalSum);
}
```

```c
#include<stdio.h>
int main(){
    int MoneyPot1;
    int @MoneyPot2;
    MoneyPot1 = 10;
    @MoneyPot2 = 30;
    int sum = MoneyPot1+@MoneyPot2;
    printf("%d\n",sum);
}
```

Invalid

```c
#include<stdio.h>
int main(){
    float percentage = 10;
    int my Marks = 20;
    printf("%d %f",my Marks,percentage);
}
```

Invalid

Valid

```c
#include<stdio.h>
int main(){
    int _house1 = 1000;
    int house2_ = 2000;
    printf("%d %d",_house1,house2_);
}
```

```
1000 2000
```

```c
#include<stdio.h>
int main(){
    int _house1 = 1000;
    int house2_ = 2000;
    printf("%d %d",_house1,house2_);
}
```

- Variables in C can have **not only data type but also storage class** that provides information about their **location and visibility**

- Will discuss later

All Primary Data Types