- Built in types
    - Int
        - 32 bit approximation of a whole number
        - Signed and unsigned
            - Unsigned is not approximation
                - Goes from 0 - 4 bill
    - Double
        - 64 bits
    - Float
        - 32 bits
    - $2^{10} = 1024$
        - Is a kilobyte kb
        - 2^{20}
            - Is a megabyte mb
        - 2^{30}
            - Is a gigabyte gb
        - 2^{40}
            - Is a terabyte tb
        - Know the first 10 powers of 2
        - $2^{26} = 64\, mb$
            - $2^{6} = 64 * 2^{20} = mb$
    - Unsigned
        - Rings of numbers
            - They cycle around
            - Abstract algebra
        - Multiplying 2 unsigned ints they could cycle around and become undefined behavior since the product is less then both of the ones being multiplied
    - Char
        - Represents low 7 bits of ASCII
            - Not signed or unsigned
        - 8 bits
            - -128 to 127
            - Or unsigned 0 - 265
    - * pointers are the set of memory addresses an abstraction
        - We represent it as
        - 0xFF
            - Equivalent 0b11111111
            - =255
                - As an int
                - These are the bit representation

          - F=15
          - 15*15=255
    - Int &
      - Reference type
      - Allies to an object
        - Bind the name to an object
        - May or may not be a pointer
        - Compiler will remove local reference since it does not need to be a pointer
      - Pointers to point at different types are distinct and disjoint sets
        - Can have int***
          - Which is a pointer to a pointer to a pointer to an int
            - Will be a 2D matrix of arrays in memory
        - You can make any boolean an int and an int to a boolean
          - True = 1
            - False = 0
          - Or true = any non zero
            - False = 0
          - Can promote int to float
            - But not other way around
          - Addresses shouldn't be converted into anything else
            - What would you do with the address as int and why
- Enums
  - These also are a set of types
    - User defined types
  - We don't care about underlying representations since they are just labels
    - We may care latter
  - Function
    - Takes input gives outputs
      - Maps inputs and maps outputs
    - Total function
      - give any input and get a valid output
    - Partial function
      - Gives only right answer with correct input
    - In is the domain and out is the range / codomain / inage
  - Card does not have a real relationship between suit and rank it more just has those things
    - No comparison like an int being subset of a float
    - **Subtyping** is the topic to look up for this
  - How to fake enums
      - If language does not have them

- Make a struct with static ints
    - Static means that the variable (or function) is not bound to (or part of) the object

        ```
        struct Suit {
                static int Hearts = 0;
                int Suit; //store one of the bol S
                Suit (int S)
                        : Suit(S)//directly initialize suit from S
                {}
        };
        ```

        - Can use like namespace
        - Related to class but not part of the object
        - This is the c++ way and immediately calls S constructor
        - Does not have guard yet for Suit S2=-1;
        - Call by Suit S1 = Suit::Hearts;
        - Change constructor to check the int

        ```
        Suit(int S)
                :Suit (S)
        {
                if(S<0 || S>3)
                        std::abort();//if it is wrong just quit
        }
        ```

            - Not designed
            - Or just catech to catch the error

                ```
                If….
                        Throw std::runtime_error("invalid");
                ```

            - But program should not resume running after it broke
            - The throw is for something is wrong with the system (runtime error)
                - We have a logic error so
                    - throw std::logic_error("invaliade");
        - Can also default to a suit
            - S=Hearts;
                - We have lots of hearts now
                - Don't ever do this
                - We are hiding a bug
                    - We don't know why the program is wrong
                    - Need real good reason to do this
                - Never use cout!!!
                    - So we don't know where the error was or if it ever gets printed out

          - Not principled approach use others things
          - Throw / abort / assert

- Abort is best way
  - Or use assert(S<0 || S>3);
    - Software engineering
    - Remove entire condition for you do it