Introduction to Programming

Program

- **Dictionary Definition:** A plan or schedule of activities, procedures, etc. to be followed
- **Computing Context:** A series of instructions that can be put into a computer in order to make it perform an operation.

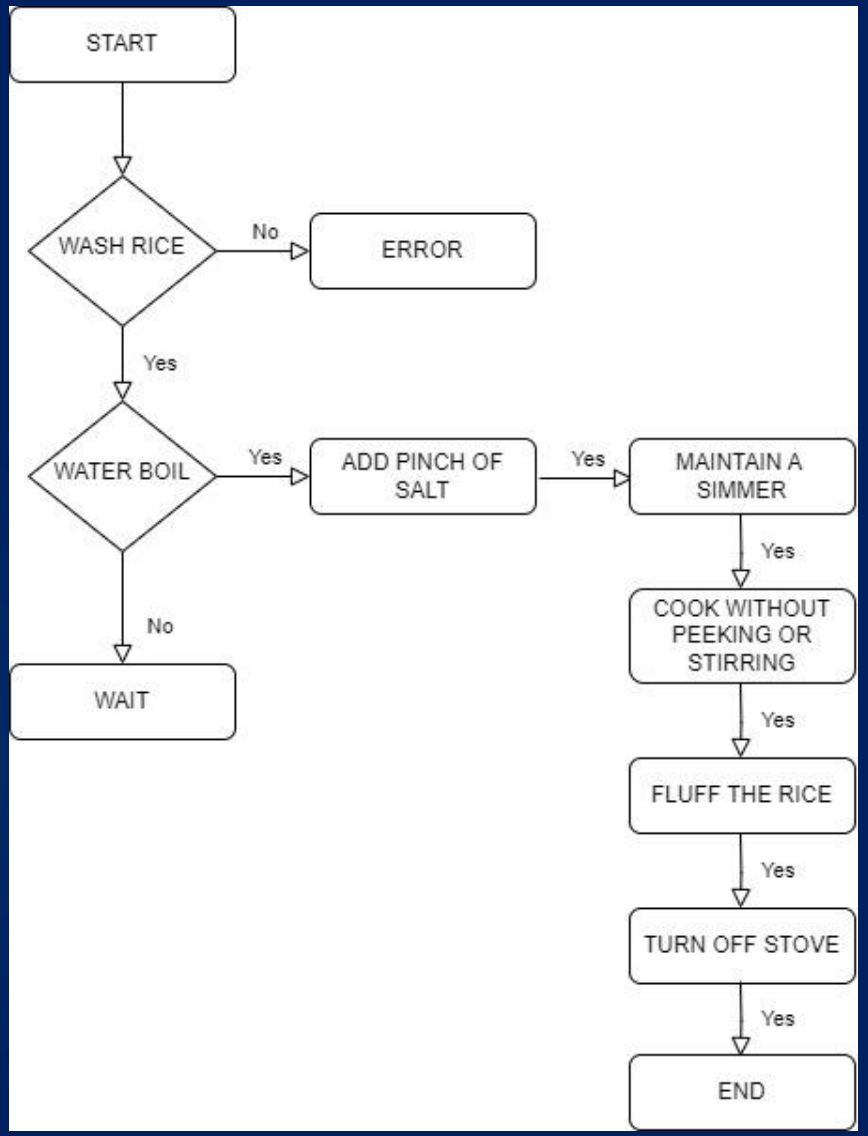
Programming

- **Programming** is writing computer code to create a program, to solve a problem.
- **Programs** are created to implement **algorithms**.
- Algorithms can be represented as **pseudocode** or a **flowchart**, and programming is the translation of these into a computer program.
  - Pseudocode: a simple way of writing programming code in English
  - Flowchart: a diagram that depicts a process, system or computer algorithm.

Pseudocode: How to cook rice

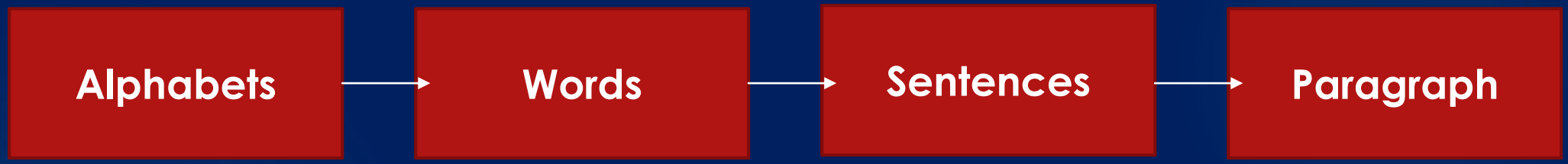
1. START
2. Wash the rice
3. Use the right ratio of water
4. Bring the water to a boil
5. Add a big pinch of salt
6. Maintain a simmer
7. Cook without peeking or stirring
8. Fluff the rice with a fork
9. Turn off the stove
10. Show rice
11. End

Flowchart: How to cook rice

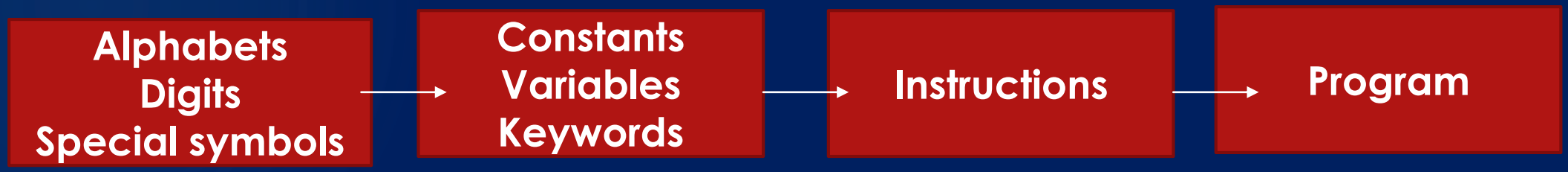


Programming

- Steps in learning English language

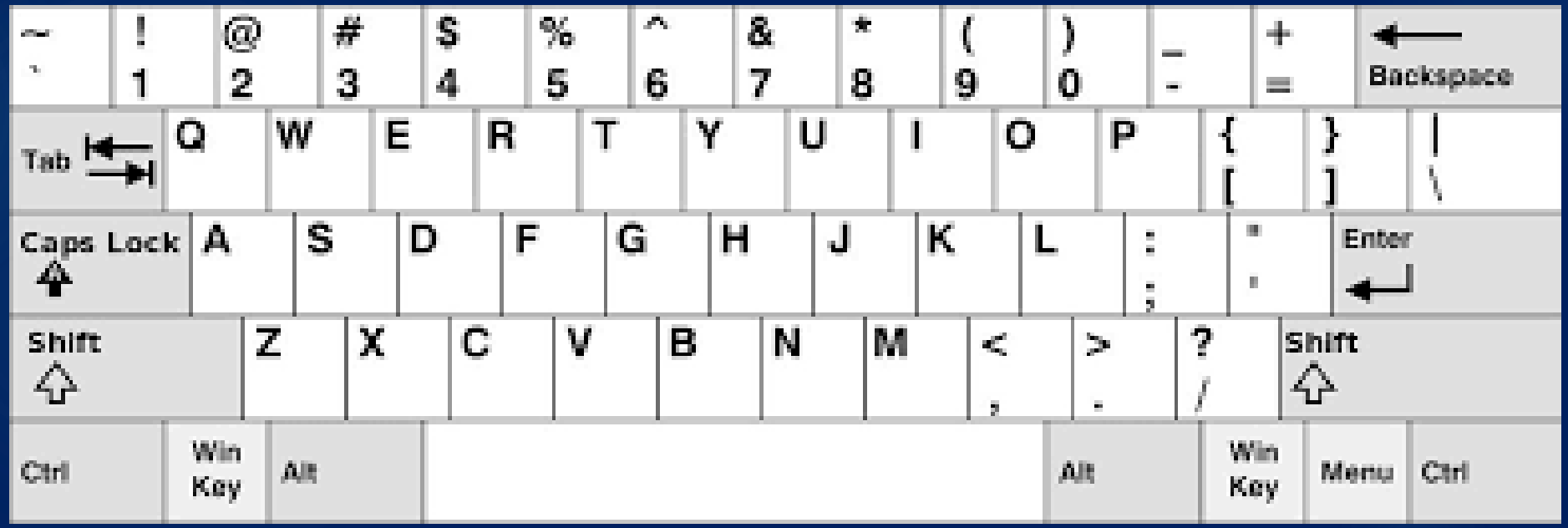


- Steps in learning C++ language

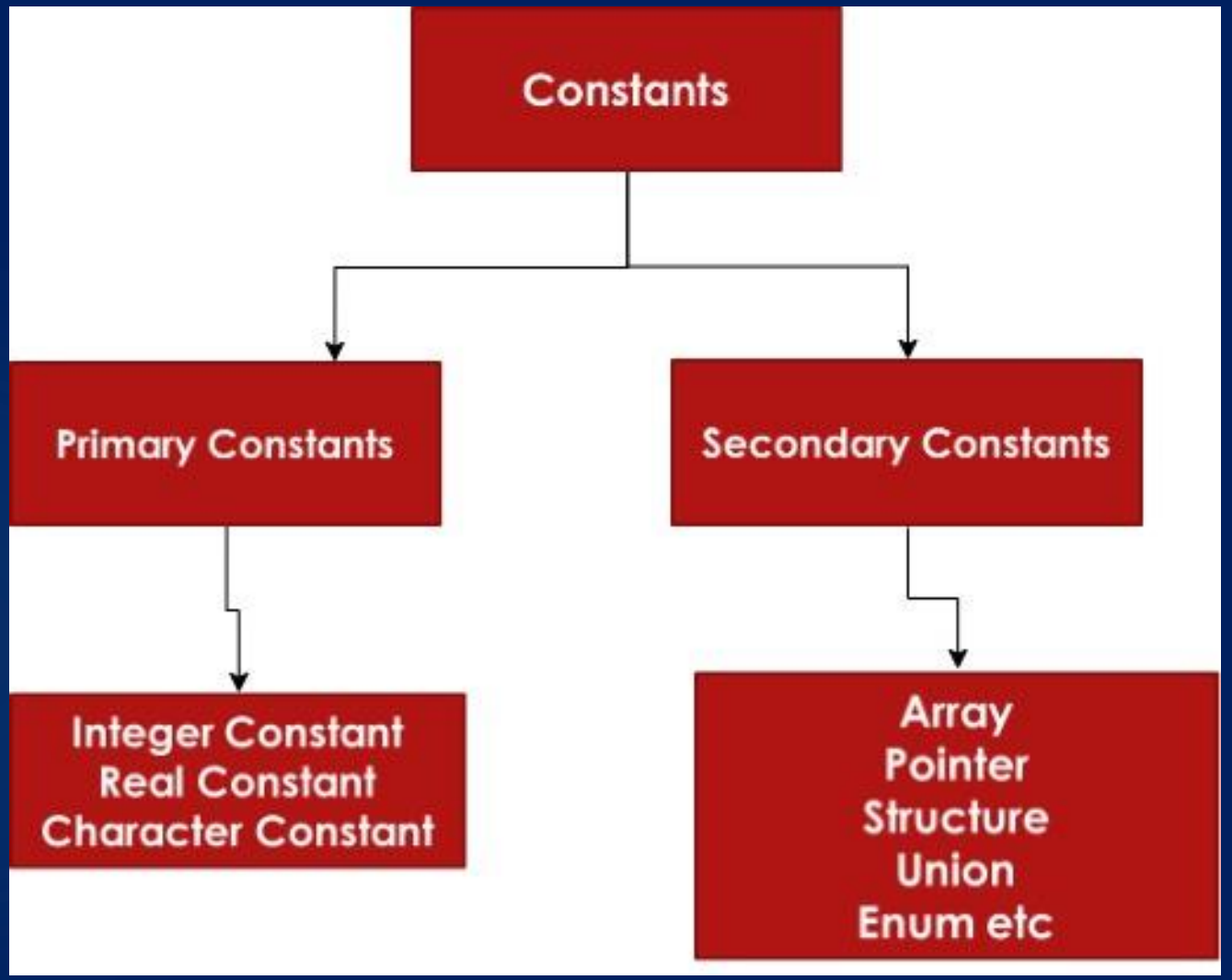


C++ language Character Set

- Alphabets
- Digits
- Special symbols



Constant, Variable, Keywords



Constant, Variable, Keywords

C++ Keyword

asm	double	new	switch
auto	else	operator	template
break	enum	private	this
case	extern	protected	throw
catch	float	public	try
char	for	register	typedef
class	friend	return	union
const	goto	short	unsigned
continue	if	signed	virtual
default	inline	sizeof	void
delete	int	static	volatile
do	long	struct	while
include			

Role of Compiler and Linker

Compiler

- A compiler is a software tool that translates high-level programming language code (like C++ or Java) into machine code that a computer can execute. It performs the following tasks:
  - Tokenization: Breaks the source code into tokens (basic elements like keywords and operators)
  - Syntax Checking: Ensures the code's structure is correct
  - Semantic Checking: Verifies the logical consistency and meaning of the code
  - Intermediate Code Generation: Creates a low-level code that is independent of the machine
  - Optimization: improves the code for better performance
  - Machine Code Generation: Converts the optimized intermediate code into machine-specific code, resulting in object files

Linker

- A linker is a tool that takes the object files generated by the compiler and combines them into a single executable program. Here's what it does:
  - Symbol Resolution: Matches function and variable names to their actual memory addresses
  - Address Assignment: Assigns final memory addresses to the code and data sections
  - Library Linking: includes additional code from libraries which are collections of pre-compiled functions
  - Relocation: Adjusts the memory addresses in the code so that everything is correctly positioned in the final executable.
  - Executable Generation: Procedures the final executable file, which can be run by the computer's operating system.

Summary

- The compiler translates and optimizes your source code into object files, while the linker combines these object files and additional libraries into a final executable program, ensuring all references are correctly resolved and positioned.