**C**

1. **Course Plan**
2. introduction to programming
3. input-output
4. operator
5. math
6. conditional statement
7. switch
8. loop
9. series
10. pattern/pyramic
11. array-one dimensional
12. array-two dimensional
13. string
14. function
15. recursion
16. file
17. structure
18. pointer
19. advance c
20. **How to read C programming**
21. translator programm
22. token
23. input-output
24. operator
25. control statement
26. array
27. function
28. **Translator Programm**
29. source code is written by higher programming language.
30. Machine code or object code is written by translator.
31. Transaltor porgramm convert source code to machine code or object code.
32. Translator programm: compiler, interpreter, assembler
33. assembler convert assembly code to machine code.
34. Compiling language: c,c++,object-c, c#, pascal, cobol,ada, visual basic,smalltalk,scheme
35. interpreting language: basic,php,python,perl,ruby, javascript
36. language processing system: source code→preprocessor→ compiler→ assembler→ linker→ binary executable
37. compiler processing system: source code→ compliling→ object code/list of errors→ input→ object code→ output.exe
38. compiler total program translate at a time then execute.
39. Interpreter line by line translate then execute.
40. Interpreter every statement make a exe file.
41. Comiler vs interpreter:

|  |  |
| --- | --- |
| **compiler** | **interpreter** |
| Whole program translate at a time. | Line by line translate. |
| Whole program error showing at a time. | Line by line showing error. |
| Compiler fast translate. | Interpret slowly translate. |
| Compiler once time compiling finished next no required translate the programm. | Every running need to translate. |
| For big computer used this. | For micro computer used this. |

1. **Algorithm**
2. algorithm means any problem solve step by step.
3. Algorithm must be easyier.
4. Every steps must be clear so that any programmer can understand.
5. Solv problem step by step.
6. Various same type of problem use this.
7. Advantages:
8. easily help to understand this programm objectives.
9. Help us to measuring errors.
10. Programm make flexibility
11. easily written any hard programm.
12. **Flowchart,pseudocode**
13. using flowchart programm can demonstrate with picture or sign.
14. Flowchart 2 types: system flowchart,programm flowchar.
15. Programm flowchart use for compose programm.
16. Algorithm vs flowchart:

|  |  |
| --- | --- |
| **algorithm** | **flowchart** |
| Using algorithm problem solve step by step. | Problem solve stages are demonstrate by diagram. |
| Algorithm describe based. | Flowchart diagram based. |
| For understanding programm need to much time. | For understanding programm need ot short time. |
| Programm flow of porcess hard to understand. | Programm flow of process easy to undersand. |
| Remove porgramm error is to hard | Remove prograam error is easy. |

1. Pseudocode word means same format of code.
2. Before writing programm any lnaguage programmer make a text based code this is called pesudocode.
3. **Stages of programm making and debuging**
4. problem selecting→ problem analysis→ programm desing→ programm development→ programm implementation→ programm maintaining
5. bug=programm errors called bug.
6. Debug= programm errors solved called debug.
7. Programm errors: syntax error, logical error information error.
8. **Introduction C Programm**
9. c invended Dennis Ritchie 1972
10. form c origin – c++, c#, java, perl, php, javascript
11. characterastics:
12. general purpose language.
13. Middle level language.
14. System programming language.
15. Structural language.
16. Platform dependent.
17. Case sensitive.
18. C use case: os,translator,editor,games,virus,anti-virus
19. **C Programm Environment**
20. editor,compilar,debugger
21. ide
22. **First C Programm**
23. header file
24. main function
25. include
26. stdio=standard input and output
27. return data type value;
28. getchar() = clean output
29. **C Comments and Escape Sequence**
30. comments: // or /\*\*/ or single line or multi line comments
31. escape sequence: \n, \t, \\
32. **C Keywords,Variables,Data Types**
33. In c programming almost 32 keywords have as a reserved word.
34. **Keywords:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Auto** | **Double** | **Int** | **struct** |
| **Break** | **Else** | **Long** | **switch** |
| **Case** | **Enum** | **Register** | **typedef** |
| **Char** | **Extern** | **Return** | **union** |
| **Const** | **Float** | **Short** | **unsigned** |
| **Continue** | **For** | **Signed** | **void** |
| **Default** | **Goto** | **Sizeof** | **volatile** |
| **Do** | **If** | **Static** | **while** |

1. Keyword use lowercase letter.
2. **Data types:**
3. char = 1 bytes
4. int = 4 bytes
5. float = 4 bytes
6. double = 8 bytes
7. memory management:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Type** | **Examples** | **Keywords** | **Storage Size** | **Format Specifier** |
| charecter | ‘a’ | char | 1 byte | %c |
| number | 125 | int | 4 bytes | %d |
| floating | 6 deimal places | float | 4 bytes | %f |
| double | 15 decimal places | double | 8 bytes | %lf |

1. **Variable:**
2. variable declaration & initialization
3. identifiers:
4. identifiers use A-Z,0-9,a-z,$,\_
5. indentifiers can’t start with number.
6. Reserved word can’t use a indentifiers.
7. Into identifiers can’t have empty space.
8. Indentifiers maximum use 32 characters but recommended 8 characters.
9. **C Input & Output**
10. input: scanf();
11. output: printf();
12. **C sizeof()**
13. sizeof()=getting memory size of this data type
14. **C ASCII Value**

|  |  |  |
| --- | --- | --- |
| **a-z** | **97-122** | **dif-32** |
| **A-Z** | **65-90** |
| **0-9** | **48-57** |  |