TA101 : Computer Programming : Comprehensive Examination

Duration : 03 Hours
Max Marks : 50
Objective Type : 15 Marks
Programming : 35 Marks

Syllabus

Variables and Data types
+ int, char, float, double
+ Declaring/Initializing/declare+initialize
+ Mathematical operators
- Binary operators (+ - / * %), Combination with = (+=, -= etc)
- Unary operators (++ --)

+ Similarities and differences between a char and int
+ Conversion of character case using airthematic
+ Applying mathematics on char and its advantages

Input & Output
- getchar(), putchar() scanf(), printf(), gets(), puts()
  Use of %d %f %c format specifiers with printf/scanf
  Use of format modifiers
    %0d %5d %-5d etc
    %7.2f %07.2f %.2f etc
Use of & (when is it not needed)

Program Control
+ Structure of the C Program
+ Use of #include , #define
+ Control statements
- if statement (with and without else)
- Nested if statements
- while loop
- for loop
- Using break and continue inside loops
- Using { } to have more lines in the control blocks

+ Logical operators
- Comparison like > < == != <= >=
- Combining logical operators with !, && and ||

Arrays

- Single dimension numeric arrays (int and float)
- Declaring and initializing arrays
- Accessing array elements
- Manipulating and using the data in arrays

- Two Dimension numeric arrays

- Strings and char Arrays
- Pointers & Strings

- Two Dimension Char arrays
  Array of strings

Built in functions:

strcpy, strcat, strncpy, strlen, strcmp, atoi, atof, strchr, strrev
strstr, toupper, tolower, itoa, abs, sin, cos, ln, pow
isalpha, islower, isupper

Modular Programming

Creating functions

- sending parameters to functions as call by value
- sending parameters as call by reference (to allow function to change values)
- sending arrays or strings to a function
- returning values from functions
- void functions
- Importance of prototype of a function
- System library functions (readymade functions)- knowledge of functions that we have used
- Inclusion of header files for certain class of readymade functions

FILES

- opening and closing files (fopen and fclose functions)
- Reading files
  fopen in "r" mode
  fgetc
  fgets
- Writing files
  fopen in "w" mode (or "a" mode)
  fputc
  fputs
- Updating files

fopen in "r+" mode
fseek
ftell

Structures

- Creating a structure (template)
- Using a structure variable in programs
  Use of . to access members of the structure
- Passing a structure variable as parameter to a function
- Returning a structure variable from a function
- Passing a structure variable as reference to a function
- Accessing members of a structure through a pointer variable
  use of -> to access the members of the structure
- Creating and using Binary files using structures
  fopen in "rb" "wb" "ab" and similar modes
  fread
  fwrite
- Updating of a datafile (fopen in "rb+" mode)

------ Last minute notes

Format specifiers

%s

```
printf("##%7s##","CAT");   ##CAT_ _ _ _##
printf("##%2s##","CAT");   ##CAT##
printf("##%-7s##","CAT");   ##_ _ _ _CAT##
```

%d

```
printf("##%d##",25);       ##25##
printf("##%5d##",25);     ##_ _ _25##
printf("##%05d##",25);     ##00025##
printf("##%1d##",25);     ##25##
```

%f

```
printf("##%f##",43.2331);  ##43.233100##
printf("##%.2f##",43.2331);  ##43.23##
printf("##%.2f##",43.2);       ##43.20##
printf("##%7.2f##",43.2331);  ##_ _43.23##
printf("##%-7.2f##",43.2331);  ##43.23_ _##
printf("##%07.2f##",43.2331);  ##0043.23##
```

1) What is the output of :

a) printf("##%-5s##","CAT");

b) printf("%09.2f",10.0/3);

2) Write the statement to input a string str using scanf

scanf("%s",str);

3) Write a program to input a string, and using pointers and predefined functions, display a count of all alphabets in it.

A CAT RAN

```
int ctr=0;
for(p=str;*p!='\0';p++)
{
 if(isalpha(*p))
  {
    ctr++;
  }
}
printf("%d",ctr);
```

4) Write a program to input a string str. Build a new string s, which is a list of alphabets that have occured in str. We have to take into account that the alphabets do not get repeated, and we only consider alphabets (no digits or special characters). This program has to be done using pointers. You may use predefined functions.

Input: A CAT RAN
Output: ACTRN

input string: str
output string: s

```
strcpy(s,"");
for(q=s,p=str;*p!='\0;p++)  {
   if(strchr(s,*p)) {
      continue;
   }
   *q=*p;
    q++;
    *q='\0';
}
puts(q);
```

1) If we want to write a function which converts a string to camel case, what would be the possible prototype of such a function.

```
void camelCase(char *)
```

2) Write a complete function definition which takes a struct student variable as a parameter, increases the marks by 10% if the marks were <50, and updates the grade accordingly. The function should return a 1 if any changes were done, otherwise it should return a zero

```
main()
{
   struct student a;
   a.marks=45;
   editMarks(&a);
   printf("%c",a.grade);
}

int editMarks(struct student *s) {
  if(s->marks<50) {
     s->marks*=110/100;
     if(s->marks>55) s->grade='B';
     else  s->grade='C';
     return 1;
  }
  else
    return 0;
}
```