RAHUL KARTHIK. S
21BEC1851

BECE320E - EMBEDDED C PROGRAMMING

DIGITAL ASSIGNMENT - 1

COMMON TO ALL SET:

1) Which of the following are invalid variable names and why?

| Variable Name | Valid or not | Reason |
|---|---|---|
| B'day | Invalid | ' is not allowed |
| int | Invalid | int is a keyword |
| $hello | Valid | $ can be used, but it's not commonly used. |
| #HASH | Invalid | # is not allowed |
| dot. | Invalid | . is not allowed |
| number | Valid | Contains only alphabets & not a keyword |
| _main() | Invalid | () is not allowed |
| temp_in_Deg | Valid | Contains alphabets & numbers and not a keyword. |
| total % | Invalid | % is not allowed |
| 1st | Invalid | Should not start with number. |
| stack-queue | Invalid | - is not allowed |
| variable name | Invalid | whitespaces aren't allowed |
| %name% | Invalid | % is not allowed |
| salary | Valid | Contains only alphabets & not a keyword. |

2) Match the following

(a) \n - Escape Sequence

RAHUL KARTHIK.S
21BEC1851

| | | |
|---|---|---|
| (b) | 3.145 | - Real Constant |
| (c) | −6513 | - Integer constant |
| (d) | 'D' | - Character Constant |
| (e) | 4.25 e-3 | - Exponential Form |
| (f) | main() | - Function |
| (g) | %f, %d, %c | - Format specifier |
| (h) | ; | - Statement Terminator |
| (i) | Constant | - Literal |
| (j) | Variable | - Identifier |
| (k) | & | - Address of Operator |
| (l) | printf() | - Output Function |
| (m) | scanf() | - Input Function |

(3) Determine which of the following are valid character constant.

(a) 'a' - Valid
(b) '$' - Valid
(c) '\n' - Valid
(d) '/n' - Invalid
(e) '\\' - valid
(f) '\a' - Valid
(g) 'T' - Valid
(h) '\0' - Valid
(i) 'xyz' - Invalid
(j) '\052' - Valid

(4) A C program contains the following statement

```
# include <stdio.h>
int i,j;
long ix;
unsigned u;
float x;  double dx;
```

RAHUL KARTHIK.S
21BEC1851

char c;

Write an appropriate printf function for each of the following situations, assuming that all integers will be displaying as decimal quantities.

(a) Display the values of i, j, x and dx, assuming that each integer quantity will have a minimum field width of four characters and each float is displayed in exponential notation with atleast 14 characters and no more than 8 decimal places.

printf("%.4d %.4d %.14.8e %.14.8e", i, j, x, dx);

(b) Repeat part (a), displaying each quantity on a seperate line.

Printf("%.4d \n %.4d\n %.14.8e\n %.14.8e\n", i, j, x, dx);

(c) Display the values of i, ix, j, x, u assuming that each integer will have a minimum field width of 5 characters, the long integer will have minimum field width of 12 characters and the float will have at least 10 characters with maximum of 5 decimal places. Do not include exponent.

printf("%.5d, %.12ld %.5d %.10.5f %5u", i, ix, j, x, u);

RAHUL KARTHIK.S
21BEC1851

(d) Repeat part (c), displaying the first three quantities on one line, following by blank line and remaining two quantities on the next line.

```
printf("%5d %12ld %5d\n\n %10.5f %5u", i, ix,
                                    j, x, u);
```

(e) Display i, u, c with minimum field of 6 characters for each integer. Place three blank spaces between each output quantity.

```
printf("%6d  %6u  %6c", i, u, c);
```

(f) Display the values of j, u, x. Display the integer question with minimum field of 5 characters. Display floating point with f-type conversion, with minimum field of 11 characters and 4 decimal points.

```
printf("%5d %5u %11.4f", j, u, x);
```

(g) Repeat (f), with each data item left justified within its respective field.

```
printf("%-5d %-5u %-11.4f", j, u, x);
```

(h) Repeat (f), with a sign (either + or —) preceding each signed data item.

```
printf("%+5d %5u %+11.4f", j, u, x);
```

RAHUL KARTHIK S
21BEC1851

(i) Repeat (f), with leading zeros filling zeros filling out the field for each of the integer quantities

```
printf("%05d %05u %011.4f", j, u, x);
```

(j) Repeat (f), with a provision for a decimal point in the value of x regardless of its value.

```
printf("% 5d % 5u % 11.4f", j, u, x);
```

(5) 4.80 :

A C program contains the following variable declaration.

```
char c1 = 'A', c2 = 'B', c3 = 'C';
```

Show the output resulting from each of the following printf statement.

(a) `printf("%c %c %c", c1, c2, c3);`

A B C

(b) `printf("%c%c%c", c1, c2, c3);`

ABC

(c) `printf("%3c %3c %3c", c1, c2, c3);`

A B C

(d) `printf("%-3c%-3c%-3c", c1, c2, c3);`

A B C

(e) `printf("c1 = %c c2 = %c c3 = %c", c1, c2, c3);`

RAHUL KARTHIK. S
21BEC1851

c1 = A  c2 = B  c3 = C

⑥ Write the first line of the function definition including the formal argument declarations, for each of the situations described below.

ⓐ A function called sample generates and returns an integer quantity.

```
int sample ()
```

ⓑ A function called 'root' accepts two integer argument and return a floating-point result.

```
float root (int n1, int n2)
```

ⓒ A function called convert accept a character and returns another character.

```
char convert (char c)
```

ⓓ A function called transfer accepts a long integer and returns a character.

```
char transfer ( long int n);
```

ⓔ A function called inverse accepts a character and returns long integer.

```
long int inverse (char ch)
```

(f) A function called process accepts an integer and two floating-point quantities and return double.

```
double process (int x, float y, float z)
```

(g) A function called value accept two double and short int. The input quantities are processed to yield double.

```
void value (double x, double y, short int
                                        z)
```

(8) write an interactive C program that accepts the name of a country as input and display the corresponding capital and vice versa. Design the program so that it executes repeatedly until the word End is entered as input.

```
#include <stdio.h>
#include <string.h>
int main() {
    int i;
    char *countries[] ={ "canada", "England",
"France", "Germany", "India", "Isreal", "Italy",
```

```
"Japan", "Mexico", "People's Republic of China"
, "Russia", "United States" };

char *capitals[] = {"Ottawa", "London",
"Paris", "Bonn", "New Delhi", "Jerusalem",
"Rome", "Tokyo", "Mexico City", "Beijing",
"Moscow", "Washington" };

char input[50];

while(1){
   printf("Enter country / Capital (or type
      'End' to exit): ");
   gets(input);
   if (strcmp(input,"End") == 0)
         break;
   int found = 0;
   for (i=0; i< 12; i++){
      if (strcmp(input, countries[i]) == 0){
         printf("The capital of %s is %s
          \n", countryries[i], capitals[i]);
         found = 1;
         break;
      }}
if (found == 0){
for (i=0; i< 12; i++){
   if (strcmp(input, capital[i]) == 0){
```

```
        printf("%s is the capital of %s\n",
            capitals[i], countries[i]);
        found = 1;
        break;
    }}}
    if (found ==0)
        printf("Not Found. Try Again");
    }
    return 0;
}
```

(9) Several declaration involving pointers are shown below. The individual declarations range from simple to complex.

```
int *p;
int *p[10];
int (*p)[10];
Int  *p(void);
int p(char *a);
int *p(char *a);
int (*p)(char *a);
int (*p(char *a))[10];
```

RAHUL KARTHIK. S
21BEC1851

```
int p(char (*a)[]);
int p(char *a[]);
int *p(char a[]);
int *p(char (*a)[]);
int *p(char *a[]);
int (*p)(char (*a)[]);
int *(*p)(char (*a)[]);
int *(*p)(char *a[]);
int (*p[10])(void);
int (*p[10])(char a);
int *(*p[10])(char a);
int *(*p[10])(char *a);
```

(10) Write an appropriate declaration for each of the following situation.

(a) `int *i, *j;`

(b) `float *fp ; double *dp;`

(c) `long *function(int a, int b);`

(d) `long function(int *a, int *b);`

(e) `float *array;`

(f) `float (*array)[30];`

(g) `char *colours[] = {"red", "green", "blue"};`

RAHUL KARTHIK. S
21BEC1851

(h) char *function (int (*func)(int));

(i) float (*func) (int, int, int)

(j) float *(*func)(int *, int *, int *);

(11) 10:46

```
char u,v = 'A';
char *pu, *pv = &v;
.....
*pv = v + 1;
u = *pv + 1;
pu = &u;
```

(a) &v is the address of v, which is F8D.

(b) pv is the pointer to v, so pv holds the address F8D.

(c) *pv is the value at the address pv points to. *pv becomes 'B', after v+1.

(d) u is assigned *pv+1, which is 'B'+1 = 'C'.

(e) &u is the address u, which is F8C.

(f) pu is pointer to u, so it holds F8C.

(g) *pu is the value at address pu which is u. *pu is 'C'.

RAHUL KARTHIK.S
21BEC1851

(12) A C Program contains the following declaration.

```
static int x[8] = {10, 20, 30, 40, 50, 60, 70, 80};
```

(a) x is name of array. In most cases, decays to the pointer to the first element of the array. This means x is equivalent to &x[0] and is type int*, pointing to first element of the array.

(b) x+2 means pointer is being performed on the array base address. This expression gives &x[2]. It moves pointer two position forward from x.

(c) *x is ~~def~~ dereferences the pointer x, which points to first element of the array, i.e. 10.

(d) *x +2, it dereferences first element which is 10, Adding 2 to 10 gives 12.

(e) *(x+2), this points &x[2], which is 30.

(13) State whether it is True or False.

1. False
2. True
3. True
4. False
5. True
6. True
7. True
8. False
9. True
10. False

(14) Write appropriate declaration and assign the given initial values for each group of variables and arrays.

(a)
```
float a = -8.2, b = 0.005;
int x = 129, y = 87, z = -22;
char c1 = 'w', c2 = '&';
```

(b)
```
double d1 = 2.88e-8, d2 = -8.4e5;
Int u = 0711;
int v = 0xffff;
```

(c)
```
long big = 123456789
double c = 0.3333333333;
char eol = '\n';
```

(d)
```
char message[] = "ERROR";
```

RAHUL KARTHIK.S
21BEC1851

(15) Write an interactive program that read positive integer number and determine whether its prime & Fibonacci Number.

```
#include <stdio.h>
int isPrime (int n) {
    int i;
    if (n<=1)  return 0;
    for (i=2; i*i <=n; i++) {
        if (n%i ==0) return 0;
    }
    return 1;
}
int isFibonacci (int n) {
    int a=0, b=1, c=a+b;
    if (n==0 || n==1) return 1;
    while (c<=n) {
        if (c==n) return 1;
        a=b;
        b=c;
        c=a+b; }
    return 0; }
int main () {
    int n;
    while (1) {
        printf("Enter Positive Number:");
```

```
        scanf("%d", &n);
        if (n==0) break;
        if (n<0) {
            printf("Enter Positive number!");
            continue; }
        if (isPrime(n)==1){
            printf("%d is Prime Number", n);
        else {
            printf("%d is not Prime Number",
                                    n);

        if (isFibonacci(n) == 1) {
            printf("%d is Fibonacci
                        Number", n);

        else
            print("%d is not Fibonacci
                        Number", n);

        printf("\n");
    }
    return 0;
}
```

(16) What are Function pointers and how are they used? Explain with example?

RAHUL KARTHIK.S
21BEC1851

Function Pointers are pointers that point to the address of the function. They allow functions to be passed as arguments to other functions, returned from functions or stored in arrays.

⍟ Passing Function as Arguments: You can pass a functions to another functions.

⍟ Arrays of Function pointers: You can create arrays of functions pointers to store and execute multiple functions based on certain conditions.

⍟ Returning a Function Pointer from a Funct-ion:

A Function can return a pointer to another functions useful in implementing state machines or function chaining.

Eg:

```
#include <stdio.h>
int add (int a, int b){
    return a+b; }
int main (){
    int (*operation)(int, int) = add;
    // Above is declaration of Func. Pointer
```

```
    int sum = operation (5, 3); // calling Func.
    printf ("Result : %d", sum);
    return 0;
}
```

(17) Write program to print the following output using for loops.

(a)
```
#include <stdio.h>
int main () {
    int i, j, rows;
    printf (" Enter number of rows : ");
    scanf ("%d", &rows);
    for (int i = 1; i <= rows ; i++ ) {
        for ( j=1; j <= i ; j++) {
            printf ("%d ", i); }
        printf ("\n"); }
    return 0;
}
```

(b)
```
#include <stdio.h>
int main () {
    int i, j,      k, rows;
    printf ("Enter number of rows:");
    scanf ("%d", &rows);
```

```
for (i = rows ; i >= 1 ; i--) {
    for (j = 1; j <= rows - i ; j++) {
        printf (" "); }
    for (k = 1 ; k <= i ; k++) {
        printf ("* "); }
    printf ("\n"); }
return 0;
}
```

(18) Determine the value of each of the following logical expression.

(a) false

(b) true

(c) true

(d) true

(e) true

(19) Write a program to print the size of various data types in C.

```
#include <stdio.h>
int main() {
    printf ("size of All Data types : ");
    printf ("char : %. zu ", sizeof (char));
```

```
printf ("Int : %zu", sizeof (int));
printf ("short : %zu", sizeof (short));
printf ("Long : %zu", sizeof (long));
printf ("Float : %zu", sizeof (float));
printf ("Double : %zu", sizeof (double));
printf ("Long Double : %zu", sizeof (long
                                double));
printf ("Void : %zu", sizeof (void));
return 0;
}
```

(20) state errors, if any, in the following input statements.

(a) `scanf ("%c %f %d", &city, &price, &year);` — & operator is used

(b) `scanf ("%s%d", city, &amount);` — & operator is used

Mentioned in within string

(c) `scanf ("%f %d", &amount, &year);`

Mentioned within strings

(d) `scanf ("\n %f", &root);` → & operator is used

(e) `scanf ("%c %d %ld", &code, &count, &Root);`

& operator is used

RAHUL KARTHIK.S
21BEC1851

⑦ Write an appropriate array definition for each of the following problem situation.

ⓐ `int c[12] = {1, 4, 7, 10, 13, 16, 19, 22, 25, 28, 31, 34};`

ⓑ `char point[] = {'N', 'O', 'R', 'T', 'H'};`

ⓒ `char letters[] = {'N', 'ES', 'E', 'W'};`

ⓓ `float consts[] = {0.005, -0.032, 106, 0.167, -0.358, 0.015};`

ⓔ `int m[3][4] = {{10, 12, 14, 16}, {20, 22, 24, 26}, {30, 32, 34, 36}};`

RAHUL KARTHIK · S
21BEC1851

SET - 2 :

① What rules must be followed when naming identifiers in C?

* It should contain only alphabets, numbers and underscore.
* It can't begin with numbers. It should begin only with alphabets and underscore.
* Keywords can't be used as identifiers.
* It should not contain other special characters and whitespaces.
* It is case-sensitive. E.g : Var and VAR are different.

Example for Valid Identifiers : hello, int123, no_of_days.

② What are the primary data types in C and their typical sizes ?

* char :- Used to store single character
  size : 1 byte
* int :- Used to store integer
  size : Short int - 2 bytes
  long int - 4 bytes
  signed or unsigned int - 2 or 4 bytes
* Float :- used to store floating point numbers.
  size : ~~siz~~ 4 bytes

★ Double :- Used to store double floating point numbers.

Size : double - 8 bytes

long double - 10 bytes

★ void :- Used for representing absence of return type, generally used in functions.

③ Give examples for type casting in C.

★ There are two types of type casting in C. They are Implicit & Explicit type casting.

★ Implicit or Automatic Type Casting : The type casting is automatically performed by the compiler.

★

E.g :

```
int n = 5;
double d = n; // Automatically type-casted
```

★ Explicit or Manual Type Casting : The type casting is done by manually mentioning the data type.

Eg :

```
int x;
double c = 10.5;
x = (int) c; // 10.5 becomes 10 and stored in x.
```

RAHUL KARTHIK. S
21BEC1851

4) With examples, briefly explain about comma, dot and arrow operators in C.

(i) Comma :

- It is a binary operator.
- It evaluates first operands from left to right order, and last expression is stored.
- It has lowest precedence among all operators.

Eg :

```
int i = (1+3, 2*5, 3-1); // i=2 is stored
```

(ii) Dot :

- The dot operator is used to access members of structure or union.
- It is used to access its individual members.

E.g.: (only code snippet, not complete code)

```
struct Point {
    int x; int y; };

int main() {
    struct Point p1 = {10, 20};
    printf("%d", p1.x);
    return 0; }
```

(iii) Arrow :

(*) It is used to access members of structure or union when you have pointers to structure and union.

(*) It is a combination of dereferencing a pointer and accessing a member in single steps.

E.g : (Code Snippet, Not a ~~full~~ complete code)

```
struct Point {
    int x ; int y ; };
int main () {
    struct Point p1 = {10, 20};
    struct Point *ptr = &p1; // ptr to
                                struct
    printf ("%d", ptr→x );
}
```

5) write a program that will determine the first n Fibonacci Number.

```
#include <stdio.h>
int main () {
    int n, i ;
    int f1 = 1, f2 = 1, fn ;
    printf ("Enter Number of Terms:");
    scanf ("%d", &n );
```

```
if (n <= 0) {
    Printf ("Enter Positive Number");
else if (n == 1)
    printf ("%.d", f1);
else {
    Printf ("%.d %.d ", f1, f2);
    for (i = 3; i <= n; i++) {
        fn = f1 + f2;
        Printf ("%.d ", fn);
        f1 = f2;
        f2 = fn;
    }
    return 0;
}
```

⑥ Using c program, calculate the cumulative project of a list of n numbers.

```
# include <stdio.h>
int main () {
    int n, i;
    float num, CP = 1.0;
    ~~scan~~ printf ("Enter the number of
                    elements : ");
    Scanf ("%d", &n);
```

```
if (n<=0){
    printf("Enter positive number");
}
else {
  for (i=1; i<=n; i++) {
      printf(
      scanf("%f", &num);
      cp *= num;
  }
}
printf("Cumulative Product: %f", cp);
return 0;
}
```

⑦ Read a 5-letter word, encode word letter-by-letter by subtracting 30 from numerical value that represents each letter. For example 'a' - 97 would become 'c' - 67.

```
#include <stdio.h>
int main(){
    char word[6];
    int i;
    printf("Enter the word:");
    scanf("%s", word);
```

```
    printf ("Encoded Words : ");
    for (i=0; i<5; i++) {
        word[i] = word[i] - 30;
        printf ("%c", &word[i]);
    }
    return 0;
}
```

⑧ Point out the errors, if any in the Program.

```
# include <stdio.h>
int main ()
{
int temp;
scanf ("%d" &temp);
/*switch is not used, instead ~~in~~ "if" is
  used, case is also not used */
if (temp <= 20)   //colon is not used
printf ("Oooooo ohhhh! Damn cool!\n");
else if (temp > 20 && temp <= 30 )
printf ("Rain rain here again! \n");
else if (temp > 30 && temp <= 40)
printf ("wish I am on Everest \n");
else    // else instead of default
```

```
printf("Good old nagpur weather\n");
return 0;
}
```

(9) What will be output of the program:

(a) 100

(b)
2
4
3
6
8
5
3
5
1

(c) 3 3 1

(d)
malayalam is a palindrome
malayalam is a palindrome
:
:

(10) What is the output of following code?

```
90807060504030201100
```

(11) What is the output of following code?

Delhi Bangalore END

(12) What is the output of the following code?

1

(13) What is the output of the following segment when executed?

-40
40n

(14) Write a C program to find the numbers of and sum of all integers greater than 100 and less than 200 that are divisible by 7.

```
#include <stdio.h>
int main () {
    int i, count = 0 , sum = 0;
    for (i = 101 ; i < 200 ; i++) {
        if (i % 7 == 0) {
            count ++ ;
            sum += i ;
        }
}

printf(" count :  %d\n ", count);
printf (" Sum : %d\n ", sum) ;
```

```
    return 0;
}
```

(15) Write a C program to print the right half pyramid of stars upto a given row number.

```
#include <stdio.h>
int main() {
    int r, i, j;
    Printf ("Enter rows : ");
    scanf ("%d", &r);
    for (i = 1; i <= r; i++) {
        for (j=1; j <= i; j++)
            printf ("* ");
        printf ("\n");
    }
    return 0;
}
```