C Fundamentals Objective: Sheet 1

Q1.
```
int main()
{
 char a = '\012';
 printf("%d", a);
 return 0;
}
```
a) 10 b) 12 c) 0 d) Compiler Error

Q2. Which of the following statements should be used to obtain a remainder after dividing 3.14 by 2.1 ?
a) remainder = 3.14 % 2.1 b) remainder = modf(3.14, 2.1)
c) remainder = fmod(3.14, 2.1)
d) remainder cannot be obtained in floating point division.

Q3.
```
#define square(x) x*x
int main()
{
  int x;
  x = 36/square(6);
  printf("%d", x);
  return 0;
}
```
a) 6 b) 36 c) 1 d) 0

Q4. How would a value 1.66 is round off to 2.0?
a) ceil(1.66) b) floor(1.66) c) roundup(1.66) d) roundto(1.66)

Q5. Which of the following is the correct order of evaluation for the expression:
```
"z = x + y * z / 4 % 2 – 1"
```
a) * / % + - = b) = * / % + - c) / * % - + = d) * % / - + =

Q6.
```
#define CUBE(x) (x*x*x)
int main()
{
   int a, b = 3;
   a = CUBE( b++);
   printf("%d, %d", a, b);
   return 0;
 }
```
a) 27, 5 b) 27, 6 c) 64, 6 d) 64, 5

Q7. Which of the following is true?
a) gets() can read a string with newline characters but a normal scanf() with %s can not.
b) gets() can read a string with spaces but a normal scanf() with %s can not.
c) gets() can always replace scanf() without any additional code.
d) none of the above

Q8.
```
int main()
{
 int x = 50;
 {
   int x = 10;
```

```
      printf("%d", x);
    }
    printf("%d", x);
    return 0;
  }
```

a) 10 10 b) 50 50 c) 10 50 d) 50 10

Q9.
```
#define SWAP(a, b) int t; t = a, a = b, b = t;
int main()
{
  int a = 10, b = 20;
  SWAP(a, b);
  printf("%d, %d", a, b);
  return 0;
}
```

a) 10, 20 b) Error: Declaration not allowed in macros.
c) 20, 10 d) Error: Undefined symbol t.

Q10.
```
int main()
{
  int i = 5, j = 10, k = 15;
  printf("%d", sizeof(k /= i+j));
  printf("%d", k);
  return 0;
}
```

a) 4 1 b) 4 15 c) 1 1 sd) 1 15

Q11.
```
int main()
{
  int i = -3, j = 2, k = 0, m;
  m = ++i && ++j && ++k;
  printf("%d, %d, %d, %d",i, j, k, m);
  return 0;
}
```

a) -2, 3, 1, 1 b) 2, 3, 1, 2 c) 1, 2, 3, 1 d) 3, 3, 1, 2

Q12. To round off value of a variable x from float to int, correct way is:
a) y = (int) (x + 0.5) b) y = int(x + 0.5) c) y = (int) x + 0.5 d) y = (int) ((int) x + 0.5

Q13.
```
#include<math.h>
int main()
{
  printf("%d, %d, %d",sizeof(3.14f),sizeof(3.14),sizeof(3.14l));
  return 0;
}
```

a) 4, 4, 4 b) 4, 8, 8 c) 4, 8, 10 d) 4, 10, 10

Q14.In C, which of the following is valid datatypes?
a) “long long double” b) “unsigned long long int”
c) “unsigned long double” d) all are valid e) all are invalid

Q15.
```
int main()
{
  printf("%x\n", -2 << 2);
  return 0;
```

```
    }
```

a) ffff b) 0 c) fff8 d) Runtime Error

Q16.
```
int main()
{
   int x = 45;
   printf("%d, %d, %d", x<=45, x=40, x>=10);
   return 0;
}
```

a) 1, 45, 1 b) 1, 45, 0 c) 1, 1, 1 d) 1, 40, 1

Q17.
```
int main()
{
   int k, num = 30;
   k = (num > 5 ? ( num <= 10 ? 100 : 200) : 500);
   printf("%d", num);
   return 0;
}
```

a) 30 b) 100 c) 200 d) 500

Q18.
```
int main()
{
   int i;
   #if A
   printf("Enter any number : ");
   scanf("%d", &i);
   #elif B
   printf("The number is odd");
   printf 0;
}
```

a) The number is odd b) Enter any number : 10 c) Garbage value
d) Compiler Error: Unexpected end of file because there is no matching #endif.

Q19.
```
int main()
{
   char ch;
   ch = 'A';
   printf("%c, ", ch >= 'A' && ch <= 'Z' ? ch + 'a' - 'A' : ch);
   printf(%c", ch >= 'A' && ch <= 'Z' ? ch : ch + 'a' - 'A');
   return 0;
}
```

a) a, A b) A, a c) Compiler Error d) Garbage Value

Q20. Suppose a, b, c and d are int variables. For ternary operator in C (? :), pick the best statement:
a) a>b ? : ; is valid statement i.e. 2nd and 3rd operands can be empty and they are implicitly replaced with non-zero value at run-time.
b) a>b ? c=10 : d=10; is valid statement i.e. based on the value of a and b, either c or d gets assigned the value of 10.
c) a>b ? (c=10, d=20) : (c=20, d=10); is valid statement i.e. based on the value of a and b, either c=10, d=20 gets executed or c=20, d=10 gets executed.
d) all of the above.

Q21.
```
int main()
{
```

```
    int i = 1;
    switch(i)
    {
      printf("Within switch block");
      case 1: printf("Case 1");
              break;
      case 2: printf("Case 2");
              break;
      default: printf("Default");
    }
    return 0;
  }
```

a) Within switch block Case 1 b) Within switch block Default c) Case 1 d) Default

Q22.
```
int main()
{
  int i = 10, j = 15;
  if( i % 2 = j % 3)
    printf("Within if block");
  return 0;
}
```

a) Compiler Error b) Within if block c) No output will be printed d) None of the above

Q23.
```
int main()
{
  int i = 0;
  i++;
  if(i <= 5)
  {
    printf("With in if block");
    exit(0);
    main();
  }
  return 0;
}
```

a) Prints With in if block only one time. b) Prints With in if block five times.
c) Compiler Error d) Infinite loop

Q24.
```
int main()
{
  int a = 100, b = 200, c;
  if( !a >= 10 )
    b = 300;
  c = 200;
  printf("b = %d, c = %d", b, c);
  return 0;
}
```

a) b = 200, c = 200 b) b = 300, c = 200
c) b = garbage value, c = 200 d) b = 300, c = garbage value

Q25.
```
int main()
{
  unsigned int i = 65535;
  while( i++ != 0)
    printf("%d ", ++i);
```

```
    return 0;
}
```

a) 0 1 2 3 4 5 65535
b) 0 2 4 6 8 65534
c) 1 3 5 7 9 65535
d) Infinite loop

Answers: 1. (a) **2.** (c) **3.** (b) **4.** (a) **5.** (a) **6.** (b) **7.** (b) **8.** (c) **9.** (c) **10.** (b) **11.** (a) **12.** (a) **13.** (c) **14.** (b) **15.** (c) **16.** (d) **17.** (a) **18.** (d) **19.** (a) **20.** (c) **21.** (c) **22.** (a) **23.** (a) **24.** (a) **25.** (d)