

CMPE/CMSE112 MIDTERM STUDY QUESTIONS

PRECEDENCE AND ASSOCIATIVITY

OPERATORS	ASSOCIATIVITY
() [] -> .	Left to right
! ++ -- + - * & (type) sizeof	Right to left (Unary)
* / %	Left to right
+ -	Left to right
< <= > >=	Left to right
== !=	Left to right
&&	Left to right
 	Left to right
?:	Right to left
= += -= *= /= %=	Right to left
,	Left to right

Q1(6 points):

Give the output of the following program:

```
#include <stdio.h>
main()
{
    int i, j, k;
    float x;

    i = 11; j = 4; k = i/j;
    printf("%d\n", k);

    x = i/j + 2.5;
    printf("%4.2f\n", x);

    k = i/j + 2.5;
    printf("%d\n", k);

    k = (float)i/j + 2.5;
    printf("%d\n", k);

    x = (float)i/j + 2.5;
    printf("%4.2f\n", x);

    x = (float)(i/j) + 2.5;
    printf("%4.2f\n", x);
}
```

2
4.50
4
5
5.25
4.50

Q3(10 pnts.):

Give the output of the following program:

```
#include <stdio.h>
main()
{
    int a=7, x=8, y=3;
    switch(a/2)
    {
        case 4: x -= y;
                printf("x = %d y = %d\n", x, y);
        case 3: x *= y;
                printf("x = %d y = %d\n", x, y);
        case 2: x /= y;
                printf("x = %d y = %d\n", x, y);
                break;
        default: x += y;
                printf("x = %d y = %d\n", x, y);
    }
}
```

x = 24	y = 3
x = 8	y = 3

Q4 (10 points):

What is the output of the following program segment for the following values of x and y which are defined to be integers?

```
if(++x && --y)
{
    --x;
    --y;
}
else if(++x)
{
    ++x;
}
else
{
    x *= 2;
    y *= 2;
}
printf("X= %d Y=%d \n", x, y);
```

<p>x = -1, y = 0, Output: X = 2 Y=0</p> <p>x = 0, y = 1, Output: X = 3 Y = 0</p> <p>x = 1, y = 0, Output: X = 1 Y = -2</p> <p>x = 1, y = 1, Output: X = 4 Y = 0</p> <p>x = -1, y = 1, Output: X = 2 Y = 1</p>

Q5(10 pnts.):

Give the output of the following program:

```
#include<stdio.h>
main()
{
    int j, k, a;

    for(a=0, j=0, k=3; j<k; j+=2, k++) a += j+k;

    printf("a=%d j=%d k=%d\n", a, j, k);

    for(a=0, j=0, k=3; j<k; j+=2, k++)
    {
        a += j+k;
        if(j+k == 6) break;
    }

    printf("a=%d\n", a);

    for(a=0, j=0, k=3; j<k; j+=2, k++)
    {
        if(j%k == 2) continue;
        a += j+k;
    }

    printf("a=%d\n", a);
}
```

<p>a= 18 j= 6 k= 6</p> <p>a= 9</p> <p>a= 12</p>

Q6(20 pnts.):

Assuming that **int i = 1, j = 3, k = 7;** is given **before each** one of the following expressions, evaluate **the value of each expression** and also give **new values of i, j, and k** after each expression. Note that **your answers must be given in the table below.**

1. **i += j != k**
2. **k = ++j+i++**
3. **!j<i**
4. **++i==j--**
5. **k-j*2>=i?j:++k**
6. **i=j>=k**
7. **--i&&++j**
8. **--i||++j**
9. **i&&(k-=7)**
10. **(float) i+k/j**

	exp	i	j	k
1	2	2	3	7
2	5	2	4	5
3	1	1	3	<u>7</u>
4	0	2	2	<u>7</u>
5	3	1	3	7
6	0	0	3	7
7	0	0	3	<u>7</u>
8	1	0	4	<u>7</u>
9	0	1	<u>3</u>	0
10	3.0	1	3	7

Q7(24 pnts.):

Fill in the blanks in the following C code segments accordingly (Do NOT add any extra lines or statements other than those provided)

- a) Write a for loop which calculates and prints the y values for the curve $y = x^2 + 2x - 8$ for all integer values of x between 3 and 8. (assume that x and y have been declared as *int* type).

```
for ( x=3; x <=8 ; x++ )
{
    y=x*x+2*x-8 ;
    printf("y=%d\n",y) ;
}
```

- b) Write a while loop which finds the sum of grades of 30 students entered by the user. The program should ignore invalid grades that may be entered by the user (i.e an invalid grade is a grade less than zero or above one hundred). Variable *count* stores the number of grades entered and variable *grade_sum* stores the sum of grades and variable *grade* stores the grade entered.

```
int count=0, grade_sum=0, grade;
while ( count < 30 )
{
    printf(" Enter a grade: \n");
    scanf("%d",&grade) ; /* enter a grade */
    if ( grade < 0 || grade >100 )
        continue ;
    count= count + 1 ;
    grade_sum= grade_sum + grade ;
}
```

- c) Write a C code using a do-while loop where the user enters integer values and the code that finds and prints the sum of numbers until the value 0 is entered. The variable *num* stores the values entered and the variable *sum* stores their sum.

```

int num, sum=0;
do
{
printf(" Enter a number: \n");

    scanf ("%d", &num) _____; /* enter a number */

    sum= sum + num _____;

} while ( num != 0 _____);

printf (" sum = %d ", sum) _____;

```

Q8(8 pnts.):

The following C code determines if an integer value *x* entered by the user is +1, -1 or zero. If *x* is +1 or -1 the sum of non_zero values (*sum_nonzero*) is incremented by 1. Otherwise the number of zeros (*sum_zero*) is incremented by 1.

```

int x, sum_nonzero=0, sum_zero=0;
if ( x == 1 || x == -1 )
    sum_nonzero++;
else if ( x == 0 ) sum_zero++;
else printf(" You entered an invalid number");

```

Fill in the blanks in the C code provided below such that the two codes are equivalent.

```

int x, sum_nonzero=0, sum_zero=0;

switch ( x )
{
    case 1 : case -1 _____ :
        sum_nonzero+ _____ ;
        break _____ ;
    case 0 _____ :
        sum_zero+ _____ ;
        break _____ ;
default:
    printf(" You entered an invalid number") _____ ;
}

```