



## Question 2

What is the output of the program below?

```
int main()
{
    unsigned char a, b, h;
    unsigned int c, d, e, f, g;

    c = 0xc8f66e32;
    d = 0x7f585e9d;
    e = 0xf6543e82;
    f = 0x0ff0f00f;
    a = 0xd9;
    h = 0x4e;
    b = a << 4;
    g = h;
    g <<= 4;

    printf("0: 0x%x\n", (c << 12));
    printf("1: 0x%x\n", (d >> 16));
    printf("2: 0x%x\n", e & f);
    printf("3: 0x%x\n", c & ~(f));
    printf("4: 0x%x\n", d | f);
    printf("5: 0x%x\n", (c ^ d) ^ c);
    printf("6: 0x%x\n", (e & 0xffff00) << 8);
    printf("7: 0x%x\n", (a ^ 0xff));
    printf("8: 0x%x\n", b);
    printf("9: 0x%x\n", g);

    return 0;
}
```

## Question 3

What is the output of the program below:

```
int main()
{
    char *x, *y, b[60];
    int i;

    x = b+4;
    y = x+6;

    for (i = 0; i < 59; i++) b[i] = 'A';
    b[59] = '\0';

    strcpy(b, "Smith");
    strcat(b, "Pearson");
    printf("0: %s\n", b);
    printf("1: %s\n", x);
    printf("2: %s\n", y);

    strcpy(b, "Schema-Gallery");
    strcpy(x, "Binky");
    printf("3: %s\n", b);
    printf("4: %s\n", y);

    strcat(y, "Tent");
    strcat(x, "Fly");
    printf("5: %s\n", b);
    printf("6: %s\n", y);

    for (i = 0; i < 50; i += 5) {
        strcpy(b+i, "Friend");
    }
    strcpy(y, "Tater");
    strcpy(b+20, "Tot");

    printf("7: %s\n", x);
    printf("8: %s\n", y+7);
    printf("9: %c\n", x[2]);
    return 0;
}
```

012345678901234567890123456789012345678901234567890123456789

## Question 4

When the following procedure is called, the value of `x` is `0x7652cfe0`, and the contents of the 64 bytes starting with `x` are as follows. I'm printing sets of four bytes in hexadecimal and in decimal.

Address	Hexadecimal	Decimal
0x7652cfe0	0x7652cff8	1985138680
0x7652cfe4	0x7652d010	1985138704
0x7652cfe8	0x7652cff4	1985138676
0x7652cfec	0x7652cffc	1985138684
0x7652cff0	0x7652cfe4	1985138660
0x7652cff4	0x7652cfec	1985138668
0x7652cff8	0x7652d008	1985138696
0x7652cffc	0x7652cff8	1985138680
0x7652d000	0x7652d004	1985138692
0x7652d004	0x7652d010	1985138704
0x7652d008	0x7652cff8	1985138680
0x7652d00c	0x7652d000	1985138688
0x7652d010	0x7652cfe4	1985138660
0x7652d014	0x7652d004	1985138692
0x7652d018	0x7652d00c	1985138700
0x7652d01c	0x7652d008	1985138696

Your job is to tell me the output of the following procedure:

```
void a(unsigned int *x)
{
    unsigned int **e, ***f;

    e = (unsigned int **) x;
    f = (unsigned int ***) x;

    printf("0: %d\n", *x);
    printf("1: %d\n", x[3]);
    printf("2: %d\n", x[14]);
    printf("3: %d\n", **e);
    printf("4: %d\n", e[4][0]);
    printf("5: 0x%x\n", e[3][2]);
    printf("6: 0x%x\n", ***f);
    printf("7: 0x%x\n", f[2][0][0]);
    printf("8: 0x%x\n", f[1][1][2]);
    printf("9: 0x%x\n", (unsigned int) (e+12));
}
```

## Question 5

When the procedure to the below is called, the value of **a** is `0x41696f40`, and the contents of the 64 bytes starting with **a** are as shown below. Each set of four bytes is printed in decimal, hexadecimal, and as four characters.

Address	Decimal	Hexadecimal	As four characters
<code>0x41696f40</code>	<code>2780278</code>	<code>0x002a6c76</code>	<code>'v'   'l'   '*'   '\0'</code>
<code>0x41696f44</code>	<code>1097428802</code>	<code>0x41696f42</code>	<code>'B'   'o'   'i'   'A'</code>
<code>0x41696f48</code>	<code>2019819640</code>	<code>0x78640078</code>	<code>'x'   '\0'   'd'   'x'</code>
<code>0x41696f4c</code>	<code>1819869285</code>	<code>0x6c790065</code>	<code>'e'   '\0'   'y'   'l'</code>
<code>0x41696f50</code>	<code>1097428840</code>	<code>0x41696f68</code>	<code>'h'   'o'   'i'   'A'</code>
<code>0x41696f54</code>	<code>1097428804</code>	<code>0x41696f44</code>	<code>'D'   'o'   'i'   'A'</code>
<code>0x41696f58</code>	<code>2037938549</code>	<code>0x79787975</code>	<code>'u'   'y'   'x'   'y'</code>
<code>0x41696f5c</code>	<code>1097428806</code>	<code>0x41696f46</code>	<code>'F'   'o'   'i'   'A'</code>
<code>0x41696f60</code>	<code>7567727</code>	<code>0x0073796f</code>	<code>'o'   'y'   's'   '\0'</code>
<code>0x41696f64</code>	<code>1097428809</code>	<code>0x41696f49</code>	<code>'I'   'o'   'i'   'A'</code>
<code>0x41696f68</code>	<code>1097428806</code>	<code>0x41696f46</code>	<code>'F'   'o'   'i'   'A'</code>
<code>0x41696f6c</code>	<code>1802531437</code>	<code>0x6b70726d</code>	<code>'m'   'r'   'p'   'k'</code>
<code>0x41696f70</code>	<code>28160</code>	<code>0x00006e00</code>	<code>'\0'   'n'   '\0'   '\0'</code>
<code>0x41696f74</code>	<code>1097428848</code>	<code>0x41696f70</code>	<code>'p'   'o'   'i'   'A'</code>
<code>0x41696f78</code>	<code>1870230784</code>	<code>0x6f797500</code>	<code>'\0'   'u'   'y'   'o'</code>
<code>0x41696f7c</code>	<code>1701904502</code>	<code>0x65710076</code>	<code>'v'   '\0'   'q'   'e'</code>

Please tell me the output of the following procedure:

```

typedef struct {
    int i;
    char b;
    char *s;
} S1;

typedef struct {
    int i;
    char b;
} S2;

typedef struct {
    int i;
    char *s;
} S3;

void pm(char *a)
{
    S1 *x;
    S2 *y;
    S3 *z;

    x = (S1 *) a;
    y = (S2 *) a;
    z = (S3 *) a;

    printf("0: %c\n", x->b);
    printf("1: %d\n", x->i);
    x++;
    printf("2: %s\n", x->s);
    printf("3: 0x%x\n", (unsigned int) &(x[1].b));

    printf("4: %c\n", y[2].b);
    printf("5: %d\n", y[5].i);
    printf("6: 0x%x\n", (unsigned int) &(y[1].b));

    printf("7: %s\n", z->s);
    printf("8: %d\n", z[2].i);
    z += 3;
    printf("9: %s\n", z->s);

    return;
}
    
```