1.

```
/* Coloquei 4 respostas, indo do menos compacto ao mais compacto. */

int ehBissexto (int ano)
{
    int resposta;

    if ((ano%400==0) || /* Divisível por 400, ou... */
        (ano%4==0 && ano%100!=0)) /* ... divisível por 4, mas não por 100. */
        resposta = 1;
    else
        resposta = 0;

    return (resposta);
}

/* Com 1 modificação: a resposta começa com um valor padrão, que só é mudado se
   necessário. */

int ehBissexto (int ano)
{
    int resposta = 0;

    if ((ano%400==0) || /* Divisível por 400, ou... */
        (ano%4==0 && ano%100!=0)) /* ... divisível por 4, mas não por 100. */
        resposta = 1;

    return (resposta);
}

/* Outra modificação: não precisamos de uma variável para a resposta, dá para
   retornar direto! */

int ehBissexto (int ano)
{
    if ((ano%400==0) || /* Divisível por 400, ou... */
        (ano%4==0 && ano%100!=0)) /* ... divisível por 4, mas não por 100. */
        return (1);

    return (0);
}

/* Mais uma modificação: não precisamos colocar a resposta em uma variável. A
 própria condição que estamos usando no if já é, ela mesma, uma expressão cujo
 resultado é 0 ou 1. Desta forma, podemos retornar diretamente o resultado da
 expressão! Na prática, esta seria a resposta mais comum. */

int ehBissexto (int ano)
{
    return ((ano % 400 == 0) || ((ano % 4 == 0) && (ano % 100 != 0)));
}

/* Nota: daria para encolher mais ainda, mas aí fica difícil de ler/entender. */
```

2.

```
/* Os números formam uma PA se a razão entre eles for sempre igual. */
int ehProgressaoAritmetica (int n1, int n2, int n3, int n4)
{
    int razao = n2 - n1;

    if (n3 - n2 == razao && n4 - n3 == razao)
        return (razao);

    return (0);
}
```

3.

```
3
15
36
66
```

Aqui tem algumas coisas que você precisa observar. Uma é o escopo das variáveis i e n: as mudanças que ocorrem dentro da função func não afetam o valor das variáveis na função main - na verdade, nem mesmo as chamadas da função se afetam umas às outras. A outra coisa é que existe um padrão: você não precisa executar manualmente cada iteração do código, na verdade, após a segunda chamada da função, você deve ser capaz de entender o que ela faz, e pode simplesmente aplicar o raciocínio às demais chamadas. A última coisa a se observar é que este programa faz uma porção de somas de termos de uma PA de forma "força bruta" e cheia de redundâncias: existem jeitos mais inteligentes de fazer isso, mas aí eu não estaria testando as outras coisas que você precisa observar. ;)