MC-102 — Aula 09
Comandos Repetitivos

Eduardo C. Xavier

Instituto de Computação – Unicamp

4 de Abril de 2017

Roteiro

1. Laços Encaixados
   - Números Primos
   - Dados
   - Mega-Sena

2. Exercícios

Laços Encaixados: Primos

- A geração de números primos é uma parte fundamental em sistemas criptográficos como os utilizados em *internetbanking*.
- Já sabemos testar se um determinado número é ou não primo.
- Imagine agora que queremos imprimir os n primeiros números primos.
- Como resolver este problema?

Laços Encaixados: Primos

- O programa abaixo verifica se o valor na variável **candidato** corresponde a um primo:

```
divisor = 2;
eprimo = 1;
while( divisor <= candidato/2 && eprimo ){
   if(candidato % divisor == 0)
      eprimo = 0;
   divisor++;
}
if(eprimo){
   printf("%d, ", candidato);
}
```

Laços Encaixados: Primos

- Criamos um laço externo e usamos uma variável contadora **primosImpressos**, que contará o número de primos impressos durante a execução deste laço.

```
while(primosImpressos < n){

   //trecho do código anterior que
   //checa se candidato é ou não é primo

   if(eprimo){
      printf("%d, ", candidato);
      primosImpressos++;
   }
   candidato++;//Testa próximo número candidato a primo
}
```

Laços Encaixados: Primos

- Incluímos uma parte inicial de código para leitura de **n** e inicialização de variáveis.
- Para finalizar, basta incluir o trecho de código que checa se um número é primo ou não.

```
int main(){
  int divisor, candidato, primosImpressos, n, eprimo;
  printf("\n Digite um número inteiro positivo:");
  scanf("%d",&n);

  candidato = 2;
  primosImpressos = 0;
  while(primosImpressos < n){

    //trecho do código que checa
    //se candidato é ou não é primo

    if(eprimo){
      printf("%d, ", candidato);
      primosImpressos++;
    }
    candidato++;//Testa próximo número candidato a primo
  }
}
```

Laços Encaixados: Primos

Código completo:

```
int main(){
  int divisor, candidato, primosImpressos, n, eprimo;

  printf("\n Digite um número inteiro positivo:");
  scanf("%d",&n);

  candidato = 2;
  primosImpressos = 0;
  while(primosImpressos < n){
    divisor = 2;
    eprimo=1;
    while( divisor <= candidato/2 && eprimo ){
      if(candidato % divisor == 0)
        eprimo = 0;
      divisor++;
    }
    if(eprimo){
      printf("%d, ", candidato);
      primosImpressos++;
    }
    candidato++;//Testa próximo número candidato a primo
  }
}
```

Laços Encaixados: Primos

- O que acontece se mudarmos a variável indicadora **eprimo** para fora do primeiro laço **while**? Faz diferença?

```
int main(){
  int divisor, candidato, primosImpressos, n, eprimo;

  printf("\n Digite um número inteiro positivo:");
  scanf("%d",&n);

  candidato = 2;
  primosImpressos = 0;
  eprimo=1;  //  ************************** Saiu do laço, faz diferença??
  while(primosImpressos < n){
    divisor = 2;
    while( divisor <= candidato/2 && eprimo ){
      if(candidato % divisor == 0)
         eprimo = 0;
      divisor++;
    }
    if(eprimo){
      printf("%d, ", candidato);
      primosImpressos++;
    }
    candidato++;//Testa próximo número candidato a primo
  }
}
```

Laços Encaixados: Primos

- O que acontece se mudarmos a variável indicadora **eprimo** para fora do primeiro laço **while**? Faz diferença?
- Resposta: Quando testarmos um **candidato** que não é primo, a variável **eprimo** será setada para **0** e nunca mais será setada para **1**.
- Logo nenhum outro **candidato** posterior será identificado como primo.

```
int main(){
  int divisor, candidato, primosImpressos, n, eprimo;

  printf("\n Digite um número inteiro positivo:");
  scanf("%d",&n);

  candidato = 2;
  primosImpressos = 0;
  eprimo=1;  //  *************************** Saiu do laço, faz diferença??
  while(primosImpressos < n){
    divisor = 2;
    while( divisor <= candidato/2 && eprimo ){
      if(candidato % divisor == 0)
         eprimo = 0;
      divisor++;
    }
    if(eprimo){
      printf("%d, ", candidato);
      primosImpressos++;
    }
    candidato++;//Testa próximo número candidato a primo
  }
```

Laços Encaixados: Primos

- Note que o número 2 é o único número par que é primo.
- Podemos alterar o programa para sempre imprimir o número 2:

```
int main(){
  int divisor, candidato, primosImpressos, n, eprimo;

  printf("\n Digite um número inteiro positivo:");
  scanf("%d",&n);

  if(n > 0){
    printf("%d, ", 2);
     .....
  }
```

Laços Encaixados: Primos

- Podemos alterar o programa para testar apenas números ímpares como candidatos a primo:

```
candidato = 3;
primosImpressos = 1;
while(primosImpressos < n){
   divisor = 2;
   eprimo=1;
   while( divisor <= candidato/2 && eprimo ){
      if(candidato % divisor == 0)
            eprimo = 0;
      divisor++;
   }
   if(eprimo){
      printf("%d, ", candidato);
      primosImpressos++;
   }
   candidato = candidato + 2;//Testa próximo número candidato a pri
}
```

Laços Encaixados: Primos

Além disso sabendo que **candidato** é sempre um número ímpar:

- Não precisamos mais testar os divisores que são pares.
- Se **candidato** é sempre um número ímpar, ele não pode ser divisível por um número par, pois seria divisível por 2 também.
- Portanto basta testar divisores ímpares.

Laços Encaixados: Primos

```
int main(){
  int divisor, candidato, primosImpressos, n, eprimo;
  printf("\n Digite um numero inteiro positivo:");
  scanf("%d",&n);
  if(n > 0){
    printf("%d, ", 2);
    candidato = 3;
    primosImpressos = 1;
    while(primosImpressos < n){
      divisor = 3;  //Primeiro divisor ímpar a ser testado
      eprimo=1;
      while( divisor <= candidato/2 && eprimo ){
        if(candidato % divisor == 0)
            eprimo = 0;
        divisor = divisor + 2;  //Demais divisores são ímpar
      }
      if(eprimo){
        printf("%d, ", candidato);
        primosImpressos++;
      }
      candidato = candidato + 2;//Testa próximo número candidato a pri
    }
  }
}
```

Laços Encaixados: Dados

Problema

Imprimir todas as possibilidades de resultados ao se jogar 4 dados de 6 faces.

- Para cada possibilidade do primeiro dado, devemos imprimir todas as possibilidades dos 3 dados restantes.
- Para cada possibilidade do primeiro e segundo dado, devemos imprimir todas as possibilidades dos 2 dados restantes....
- Você consegue pensar em uma solução com laços aninhados?

Laços Encaixados: Dados

```
int main(){
  int d1, d2, d3, d4;

  printf("\nD1  D2  D3  D4\n");
  for(d1 = 1; d1 <= 6; d1++)
    for(d2 = 1; d2 <= 6; d2++)
      for(d3 = 1; d3 <= 6; d3++)
        for(d4 = 1; d4 <= 6; d4++)
          printf("%d   %d   %d   %d\n",d1,d2,d3,d4);

}
```

Laços Encaixados: Mega-Sena

- Na Mega-Sena, um jogo consiste de 6 números distintos com valores entre 1 e 60.

Problema

Imprimir todos os jogos possíveis da Mega-Sena.

Laços Encaixados: Mega-Sena

- Partimos da mesma idéia dos dados: gerar todos os possíveis valores para cada um dos 6 números do jogo.

```
int main(){
  int d1, d2, d3, d4, d5, d6;

  for(d1 = 1; d1 <= 60; d1++)
    for(d2 = 1; d2 <= 60; d2++)
      for(d3 = 1; d3 <= 60; d3++)
        for(d4 = 1; d4 <= 60; d4++)
          for(d5 = 1; d5 <= 60; d5++)
            for(d6 = 1; d6<= 60; d6++)
                printf("%d, %d, %d, %d, %d, %d\n",d1,d2,d3,d4,d5,d6);

}
```

- Qual a saída deste programa? Ele está correto?

Laços Encaixados: Mega-Sena

```
int main(){
  int d1, d2, d3, d4, d5, d6;

  for(d1 = 1; d1 <= 60; d1++)
    for(d2 = 1; d2 <= 60; d2++)
      for(d3 = 1; d3 <= 60; d3++)
        for(d4 = 1; d4 <= 60; d4++)
          for(d5 = 1; d5 <= 60; d5++)
            for(d6 = 1; d6<= 60; d6++)
                printf("%d, %d, %d, %d, %d, %d\n",d1,d2,d3,d4,d5,d6);

}
```

- As primeiras linhas impressas por este programa serão:

```
1, 1, 1, 1, 1, 1
1, 1, 1, 1, 1, 2
1, 1, 1, 1, 1, 3
1, 1, 1, 1, 1, 4
1, 1, 1, 1, 1, 5
1, 1, 1, 1, 1, 6
1, 1, 1, 1, 1, 7
1, 1, 1, 1, 1, 8
1, 1, 1, 1, 1, 9
```

Laços Encaixados: Mega-Sena

- O programa anterior repete números, portanto devemos remover repetições.

```
int main(){
  int d1, d2, d3, d4, d5, d6;

  for(d1 = 1; d1 <= 60; d1++)
    for(d2 = 1; d2 <= 60; d2++)
      for(d3 = 1; d3 <= 60; d3++)
        for(d4 = 1; d4 <= 60; d4++)
          for(d5 = 1; d5 <= 60; d5++)
            for(d6 = 1; d6<= 60; d6++)
              if( (d1!=d2) && (d1!=d3) &&.........)
                printf("%d, %d, %d, %d, %d, %d\n",d1,d2,d3,d4,d5,d6);

}
```

- Após incluir todos os testes para garantir que os números são distintos, temos a solução?

Laços Encaixados: Mega-Sena

- Não temos uma solução válida, pois o programa irá imprimir jogos como:

```
12, 34, 8, 19, 4, 45
34, 12, 8, 19, 4, 45
34, 12, 19, 8, 4, 45
```

- Todos estes jogos são um único jogo: 4, 8, 12, 19, 34, 45.
- Podemos assumir que um jogo é sempre apresentado com os números em ordem crescente.
- Dado que fixamos o valor de **d1**, **d2** necessariamente é maior que **d1**. Após fixar **d1** e **d2**, **d3** deve ser maior que **d2** etc.

Laços Encaixados: Mega-Sena

Solução correta:

```
int main(){
  int d1, d2, d3, d4, d5, d6;

  for(d1 = 1; d1 <= 60; d1++)
    for(d2 = d1 + 1; d2 <= 60; d2++)
      for(d3 = d2 +1; d3 <= 60; d3++)
        for(d4 = d3 +1; d4 <= 60; d4++)
          for(d5 = d4 +1; d5 <= 60; d5++)
            for(d6 = d5 +1; d6<= 60; d6++)
              printf("%d, %d, %d, %d, %d, %d\n",d1,d2,d3,d4,d5,d6);

}
```

Exercício

- Faça um programa que leia um número n e imprima n linhas na tela com o seguinte formato (exemplo se $n = 6$):

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
```

Exercício

- Faça um programa que leia um número n e imprima n linhas na tela com o seguinte formato (exemplo se $n = 6$):

```
+ * * * * *
* + * * * *
* * + * * *
* * * + * *
* * * * + *
* * * * * +
```

Exercício

- Um jogador da Mega-Sena é supersticioso, e só faz jogos em que o primeiro número do jogo é par, o segundo é ímpar, o terceiro é par, o quarto é ímpar, o quinto é par e o sexto é ímpar. Faça um programa que imprima todas as possibilidades de jogos que este jogador supersticioso pode jogar.