SSC 0721 – Teste e Inspeção de Software

Teste Estrutural – GFC

Prof. Marcio E. Delamaro

delamaro@icmc.usp.br

O que é o teste estrutural

- ✓ Supõe a existência de uma implementação
- ✓ Ela é analisada
- ✓ Estruturas de interesse devem ser exercitadas
- ✓ Comandos, desvios, “caminhos”

O Grafo de fluxo de controle

- ✓ Abstração de uma unidade: função, método ou procedimento
- ✓ Representa as possíveis sequências de execução dos comandos dessa unidade
- ✓ GFC $G = \langle N, E, s \rangle$ é um grafo direcionado
  - ⋆ N é o conjunto de vértices
  - ⋆ E é o conjunto de arestas
  - ⋆ $s \in N$ é o vértice inicial

GFC – conjunto de vértices

- Cada vértice representa um bloco sequencial indivisível de comandos
  - S_1, S_2, S_3, S_4, S_5
  - Sempre que S_1 é executado, os demais comandos são executados nessa sequêncial
  - Para que S_{i+1} seja executado, S_i tem que ser executado imediatamente antes

Vértices – exemplo

```
public static void bubbleSort(int[] x) {
    int n = x.length;
    for (int last = 1; last < n; last++)
    {
        for (int i = 0; i < n - last;  i++)
        {
            if (x[i] > x[i+1]) {
                int aux = x[i];
                x[i] = x[i+1];
                x[i+1] = aux;
            }
        }
    }
 }
```

Vértices – exemplo

```
public static void bubbleSort(int[] x) {
   int n = x.length;                    /* 1 */
   for (int last = 1;                   /* 1 */
        last < n;                       /* 2 */
        last++)  {                      /* 3 */
       for (int i = 0;                  /* 4 */
            i < n - last                /* 5 */
            i++) {                      /* 6 */
           if (x[i] > x[i+1]) {         /* 7 */
               int aux = x[i];          /* 8 */
               x[i] = x[i+1];           /* 8 */
               x[i+1] = aux;            /* 8 */
           }
       }
   }
}                                       /* 9 */
```

Vértices – exemplo

```
public static void bubbleSort(int[] x) {
   int n = x.length;                    /* 1 */ <--
   for (int last = 1;                   /* 1 */ <--
        last < n;                       /* 2 */ <--
        last++)  {                      /* 3 */
       for (int i = 0;                  /* 4 */
            i < n - last                /* 5 */
            i++) {                      /* 6 */
           if (x[i] > x[i+1]) {         /* 7 */
               int aux = x[i];          /* 8 */
               x[i] = x[i+1];           /* 8 */
               x[i+1] = aux;            /* 8 */
           }
       }
   }
}                                       /* 9 */ <--
```

Vértices inicial/finais

- Todo GFC possui um único vértice inicial
- É aquele que contém o comando de entrada da unidade
- Pode possuir também vértices de saída
- São aqueles que não possuem sucessores
- Correspondem aos comandos de retorno da unidade

Arestas

- Se é possível executar-se o primeiro comando de um vértice j imediatamente após o último comando de um vértice i, então existe uma aresta (i, j) no GFC.

Arestas – exemplo

```
public static void bubbleSort(int[] x) {
   int n = x.length;                    /* 1 */
   for (int last = 1;                   /* 1 --> ??? */
        last < n;                       /* 2 */
        last++)  {                      /* 3 */
       for (int i = 0;                  /* 4 */
            i < n - last                /* 5 */
            i++) {                      /* 6 */
           if (x[i] > x[i+1]) {         /* 7 */
               int aux = x[i];          /* 8 */
               x[i] = x[i+1];           /* 8 */
               x[i+1] = aux;            /* 8 */
           }
       }
   }
}                                       /* 9 */
```

Arestas – exemplo

```
public static void bubbleSort(int[] x) {
   int n = x.length;                    /* 1 */
   for (int last = 1;                   /* 1 --> 2 */
        last < n;                       /* 2 */
        last++)  {                      /* 3 */
       for (int i = 0;                  /* 4 */
            i < n - last                /* 5 */
            i++) {                      /* 6 */
           if (x[i] > x[i+1]) {         /* 7 */
               int aux = x[i];          /* 8 */
               x[i] = x[i+1];           /* 8 */
               x[i+1] = aux;            /* 8 */
           }
       }
   }
}                                       /* 9 */
```

Arestas – exemplo

```
public static void bubbleSort(int[] x) {
   int n = x.length;                    /* 1 */
   for (int last = 1;                   /* 1 */
        last < n;                       /* 2 --> 4, 9 */
        last++)  {                      /* 3 */
       for (int i = 0;                  /* 4 */
            i < n - last                /* 5 */
            i++) {                      /* 6 */
           if (x[i] > x[i+1]) {         /* 7 */
               int aux = x[i];          /* 8 */
               x[i] = x[i+1];           /* 8 */
               x[i+1] = aux;            /* 8 */
           }
       }
   }
}                                       /* 9 */
```

Arestas – exemplo

```
public static void bubbleSort(int[] x) {
   int n = x.length;                    /* 1 */
   for (int last = 1;                   /* 1 */
        last < n;                       /* 2 */
        last++)  {                      /* 3 --> 2 */
       for (int i = 0;                  /* 4 */
            i < n - last                /* 5 */
            i++) {                      /* 6 */
           if (x[i] > x[i+1]) {         /* 7 */
               int aux = x[i];          /* 8 */
               x[i] = x[i+1];           /* 8 */
               x[i+1] = aux;            /* 8 */
           }
       }
   }
}                                       /* 9 */
```

Arestas – exemplo

```
public static void bubbleSort(int[] x) {
   int n = x.length;                    /* 1 */
   for (int last = 1;                   /* 1 */
        last < n;                       /* 2 */
        last++)  {                      /* 3 */
       for (int i = 0;                  /* 4 --> 5 */
            i < n - last                /* 5 */
            i++) {                      /* 6 */
           if (x[i] > x[i+1]) {         /* 7 */
               int aux = x[i];          /* 8 */
               x[i] = x[i+1];           /* 8 */
               x[i+1] = aux;            /* 8 */
           }
       }
   }
}                                       /* 9 */
```

Arestas – exemplo

```
public static void bubbleSort(int[] x) {
   int n = x.length;                    /* 1 */
   for (int last = 1;                   /* 1 */
        last < n;                       /* 2 */
        last++)  {                      /* 3 */
       for (int i = 0;                  /* 4 */
            i < n - last                /* 5 --> 7, 3 */
            i++) {                      /* 6 */
           if (x[i] > x[i+1]) {         /* 7 */
               int aux = x[i];          /* 8 */
               x[i] = x[i+1];           /* 8 */
               x[i+1] = aux;            /* 8 */
           }
       }
   }
}                                       /* 9 */
```

Arestas – exemplo

```
public static void bubbleSort(int[] x) {
   int n = x.length;                    /* 1 */
   for (int last = 1;                   /* 1 */
        last < n;                       /* 2 */
        last++)  {                      /* 3 */
       for (int i = 0;                  /* 4 */
            i < n - last                /* 5 */
            i++) {                      /* 6 --> 5 */
           if (x[i] > x[i+1]) {         /* 7 */
               int aux = x[i];          /* 8 */
               x[i] = x[i+1];           /* 8 */
               x[i+1] = aux;            /* 8 */
           }
       }
   }
}                                       /* 9 */
```

Arestas – exemplo

```
public static void bubbleSort(int[] x) {
   int n = x.length;                        /* 1 */
   for (int last = 1;                       /* 1 */
        last < n;                           /* 2 */
        last++)  {                          /* 3 */
       for (int i = 0;                      /* 4 */
            i < n - last                    /* 5 */
            i++) {                          /* 6 */
           if (x[i] > x[i+1]) {             /* 7 --> 8, 6 */
               int aux = x[i];              /* 8 */
               x[i] = x[i+1];               /* 8 */
               x[i+1] = aux;                /* 8 */
           }
       }
   }
}                                           /* 9 */
```

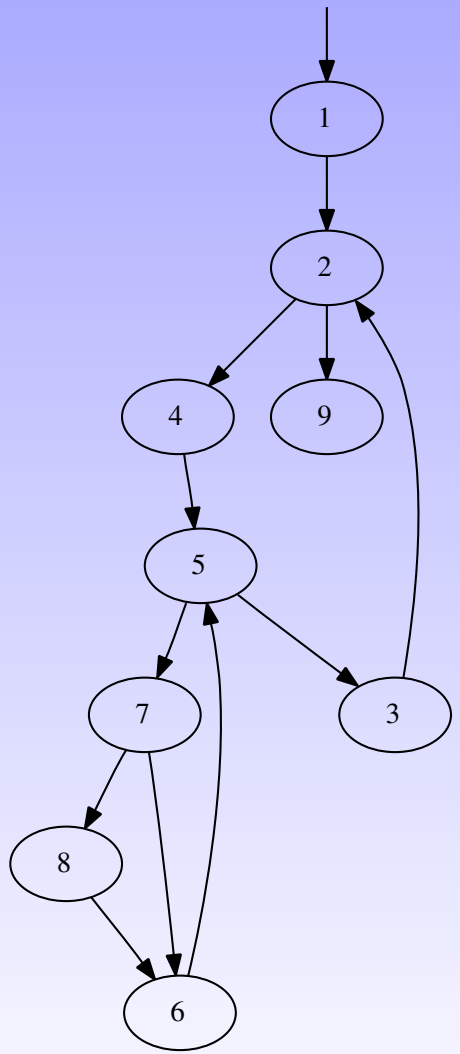
Arestas – exemplo

```
public static void bubbleSort(int[] x) {
   int n = x.length;                    /* 1 */
   for (int last = 1;                   /* 1 */
        last < n;                       /* 2 */
        last++)  {                      /* 3 */
       for (int i = 0;                  /* 4 */
            i < n - last                /* 5 */
            i++) {                      /* 6 */
           if (x[i] > x[i+1]) {         /* 7 */
               int aux = x[i];          /* 8 */
               x[i] = x[i+1];           /* 8 */
               x[i+1] = aux;            /* 8 --> 6 */
           }
       }
   }
}                                       /* 9 */
```

GFC – exemplo

1 é o nó inicial

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



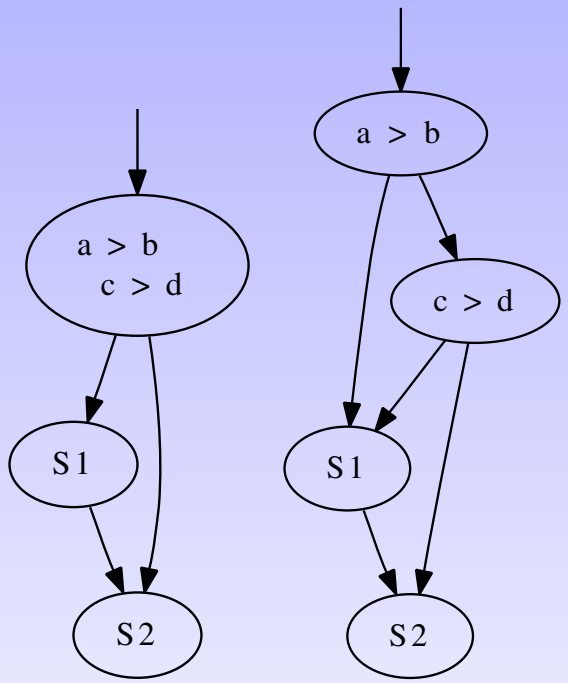
Modelo de fluxo de controle

- A construção do GFC depende de uma interpretação do código.
- Depende da linguagem do programa sendo analisado
- Depende também da decisão de como cada estrutura da linguagem deve ser considerada
- O modelo descreve como uma unidade deve ser abstraída na forma de um grafo
- Uma ferramenta é a implementação do modelo de fluxo de controle que escolhemos.

MFC – algumas questões

- Condições compostas

```
if (a > b || c > d )
```



- Influência na aplicação dos critérios

MFC – algumas questões

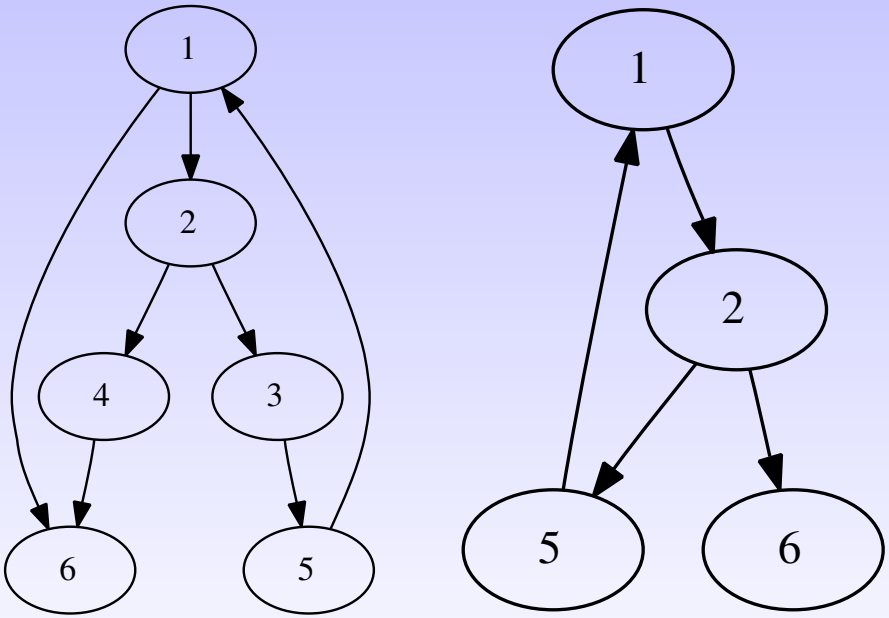
- Chamadas de subrotinas terminam um bloco de comandos?
- Como tratar chamadas de funções?
- `x = f(a) + g(b + h(c))`
  - Complexidade do GFC
  - Dificuldade de visualização
- Influência na instrumentação
  - Pré e pós-dominadores

MFC – algumas questões

- Exceções
- Num bloco, alguns comandos podem gerar exceções e outros não
- Não é facil (impossível) identificar quais comandos geram exceções
- Todos compartilham o mesmo tratador
- O tipo de fluxo (aresta) é o mesmo?

MFC – fonte × executável

```
while (true) { /*1*/
    if ( a > b) /*2*/
        S1;     /*3*/
    else
        break;  /*4*/
    S2;         /*5*/
}
S3;             /*6*/
```



Conceitos

- Um caminho é uma sequência de vértices que são ligados por arestas. No exemplo (4, 5, 7, 6, 5, 3, 2) é um caminho.
- Um caminho completo é aquele que inicia no vértice de entrada e termina em um vértice de saída. Por exemplo: (1, 2, 9) ou (1, 2, 4, 5, 3, 2, 9)
  - Cada caminho completo no GFC corresponde a um conjunto de possíveis execuções da unidade
  - Em alguns casos esse conjunto pode ser vazio pois pode não existir dado de entrada que leve à execução do caminho.
- Um caminho (completo ou não) é dito não executável quando não existe nenhum dado no domínio de entrada do programa que leve à sua execução.

Exercícios (1)

```
15  void insercao(int a[], int size) {
16      int i, j, aux;
17      for (i = 1; i < size; i++) {
18          aux = a[i];
19          j = i - 1;
20          while (j >= 0 && a[j] >= aux) {
21              a[j + 1] = a[j];
22              j--;
23          }
24          a[j + 1] = aux;
25      }
26  }
```

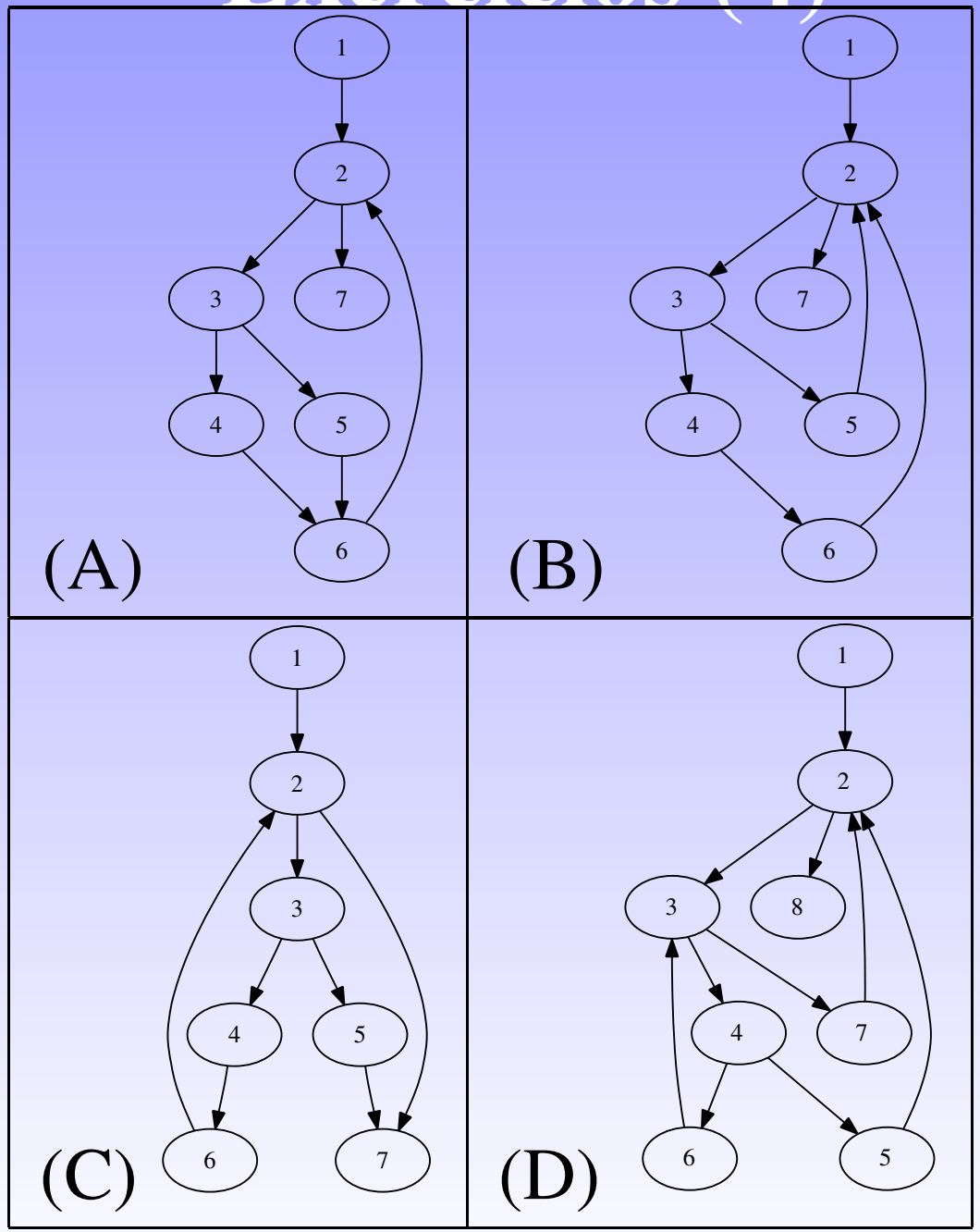
Exercícios (2)

```
28      public static void heapsort(int n, double ra[]) {
29          int l, j, ir, i;
30          double rra;
31
32          l = (n >> 1) + 1;
33          ir = n;
34          for (;;) {
35              if (l > 1) {
36                  rra = ra[--l];
37              } else {
38                  rra = ra[ir];
39                  ra[ir] = ra[1];
40                  if (--ir == 1) {
41                      ra[1] = rra;
42                      return;
43                  }
44              }
45              i = l;
46              j = l << 1;
47              while (j <= ir) {
48                  if (j < ir && ra[j] < ra[j + 1]) {
49                      ++j;
50                  }
51                  if (rra < ra[j]) {
52                      ra[i] = ra[j];
53                      j += (i = j);
54                  } else {
55                      j = ir + 1;
56                  }
57              }
58              ra[i] = rra;
59          }
60      }
```

Exercícios (3)

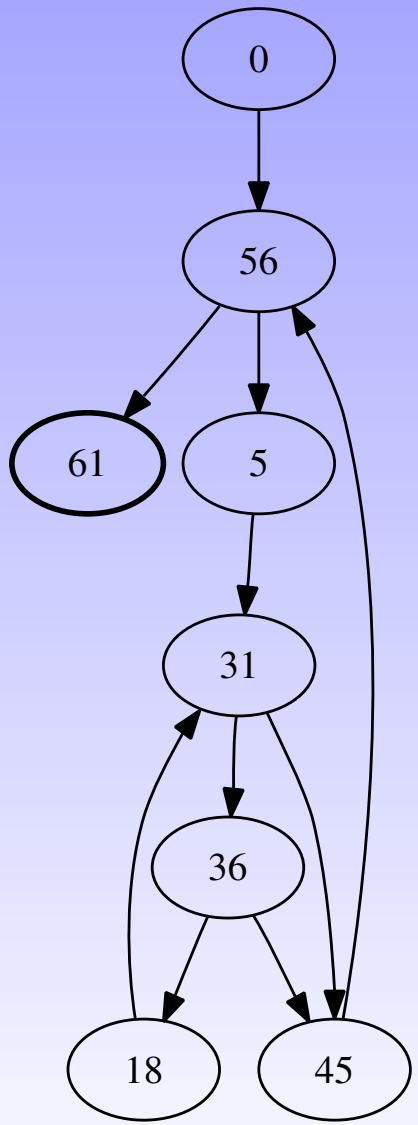
```
62      void quicksort(int a[], int lo0, int hi0) {
63          int lo = lo0;
64          int hi = hi0;
65          int mid;
66
67          // pause for redraw
68          if (hi0 > lo0) {
69              mid = a[(lo0 + hi0) / 2];
70
71              while (lo <= hi) {
72                  while ((lo < hi0) && (a[lo] < mid))
73                      ++lo;
74
75                  while ((hi > lo0) && (a[hi] > mid))
76                      --hi;
77
78                  if (lo <= hi) {
79                      swap(a, lo, hi);
80                      ++lo;
81                      --hi;
82                  }
83              }
84
85              if (lo0 < hi)
86                  quicksort(a, lo0, hi);
87
88              if (lo < hi0)
89                  quicksort(a, lo, hi0);
90
91          }
92      }
```

Exercícios (4)



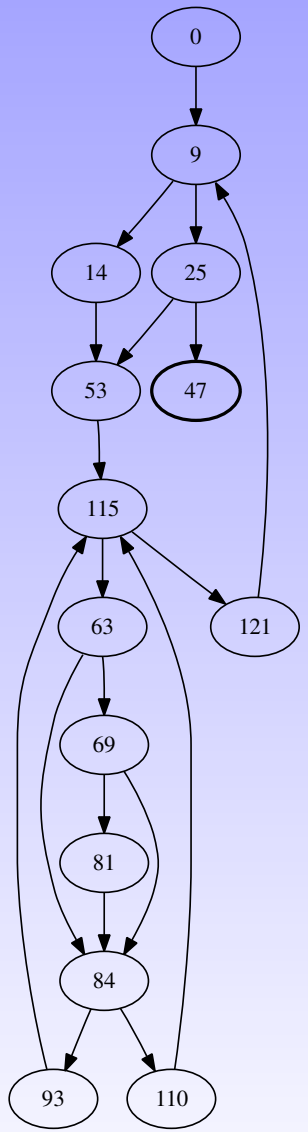
Solução (1)

```
15    void insercao(int a[], int size) {
16        int i, j, aux;
17        for (i = 1; i < size; i++) {
18            aux = a[i];
19            j = i - 1;
20            while (j >= 0 && a[j] >= aux) {
21                a[j + 1] = a[j];
22                j--;
23            }
24            a[j + 1] = aux;
25        }
26    }
```



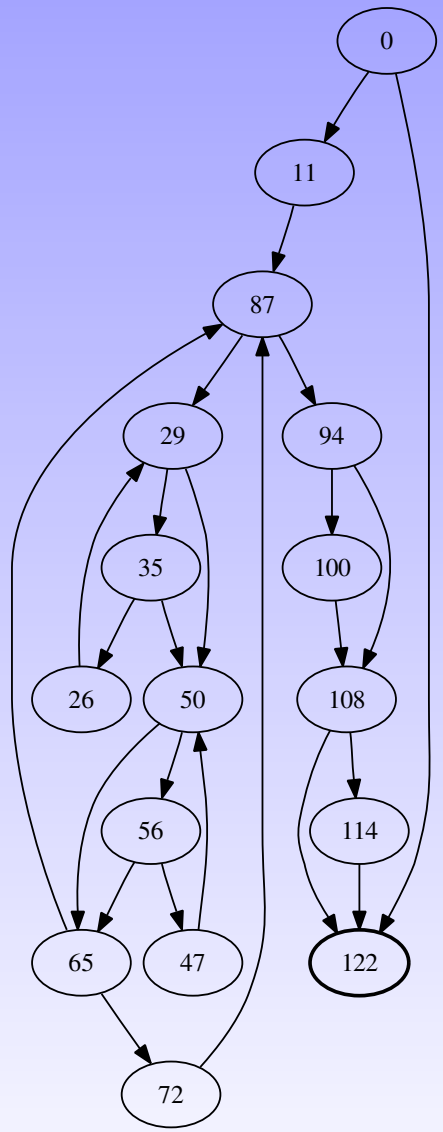
Solução (2)

```
28      public static void heapsort(int n, double ra[]) {
29          int l, j, ir, i;
30          double rra;
31
32          l = (n >> 1) + 1;
33          ir = n;
34          for (;;) {
35              if (l > 1) {
36                  rra = ra[--l];
37              } else {
38                  rra = ra[ir];
39                  ra[ir] = ra[1];
40                  if (--ir == 1) {
41                      ra[1] = rra;
42                      return;
43                  }
44              }
45              i = l;
46              j = l << 1;
47              while (j <= ir) {
48                  if (j < ir && ra[j] < ra[j + 1]) {
49                      ++j;
50                  }
51                  if (rra < ra[j]) {
52                      ra[i] = ra[j];
53                      j += (i = j);
54                  } else {
55                      j = ir + 1;
56                  }
57              }
58              ra[i] = rra;
59          }
60      }
```



Solução (3)

```
62      void quicksort(int a[], int lo0, int hi0) {
63          int lo = lo0;
64          int hi = hi0;
65          int mid;
66
67          // pause for redraw
68          if (hi0 > lo0) {
69              mid = a[(lo0 + hi0) / 2];
70
71              while (lo <= hi) {
72                  while ((lo < hi0) && (a[lo] < mid))
73                      ++lo;
74
75                  while ((hi > lo0) && (a[hi] > mid))
76                      --hi;
77
78                  if (lo <= hi) {
79                      swap(a, lo, hi);
80                      ++lo;
81                      --hi;
82                  }
83              }
84
85              if (lo0 < hi)
86                  quicksort(a, lo0, hi);
87
88              if (lo < hi0)
89                  quicksort(a, lo, hi0);
90
91          }
92      }
```



Solução (4)

```
void m(int a, int b) {
   int i = 0;
   while (i < a + b) {
      if ( i % 2 == 0 )
          i++;
      else
          i += 3;
      f(i);
   }
}
```

(A)

```
void m(int a, int b) {
   int i = 0;
   while (i < a + b) {
      if ( i % 2 == 0 )
          i++;
      else
          continue;
      f(i);
   }
}
```

(B)

Solução (4)

```
void m(int a, int b) {
    int i = 0;
    while (i < a + b) {
        if ( i % 2 == 0 )
            i++;
        else
            break;
        f(i);
    }
}
```

(C)

```
void m(int a, int b) {
    int i = 0, j = 0;
    loop1:
    while (i < a + b) {
        while ( j < a + b) {
            if ( i + j > a )
                continue loop1;
            j++;
        }
        i++
    }
    return;
}
```

(D)