
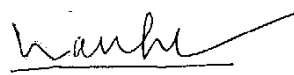


TRƯỜNG ĐẠI HỌC NHA TRANG KHOA CÔNG NGHỆ THÔNG TIN Người ra đề  Nguyễn Đức Thuận	ĐỀ THI KẾT THÚC MÔN HỌC MÔN: CẤU TRÚC DỮ LIỆU & GT Thời gian: 60 phút Lớp: 62 CNTT-4 <i>Không sử dụng tài liệu</i> ĐỀ I Ngày thi: 11 /01/2022	DUYỆT ĐỀ  Nguyễn Đức Thuận
---	---	---

- **Đề này dành cho các sinh viên ký số cuối trong Mã số sinh viên là số lẻ.**

Câu I: (2.0đ) Đánh giá độ phức tạp của đoạn chương trình sau:

```
int PT (int n) {
    if (n==0) return 0;
    for (int i =1; i <= n; i++)
        for(int j=1; j<=n; j++)
            { printf("*");
              break;}}
```

Câu II: (3.0đ)

- a. (1.5đ) Xây dựng cây nhị phân tìm kiếm lần lượt từ các giá trị sau, trong quá trình xây dựng tại nút thứ i cây không phải là cây AVL thì cân chỉnh lại cây:

12 45 40 10 7 9

- b. (1.5đ) Trình bày các bước sắp xếp dãy số sau bằng thuật toán Quick-Sort.

15 20 60 45 30 42 78 10 25 50 17 62



Câu III: (5.0đ)

Viết chương trình thực hiện các việc sau:

- (0.5đ) Nhập vào các số nguyên từ bàn phím. Kết thúc khi gõ giá trị 0. Các số nhập vào được tổ chức lưu trữ thành một cây nhị phân tìm kiếm có địa chỉ nút gốc là **goc**.
- (0.5đ) Duyệt cây nhị phân tìm kiếm theo thứ tự LNR.
- (1.0đ) Tính tổng các nút lá mang giá trị chẵn.
- (1.0đ) Viết hàm sao chép cây nhị phân có địa chỉ nút gốc là **goc** thành một cây nhị phân có địa chỉ nút gốc là **goc1**.
- (1.0đ) Viết hàm trả về giá trị của nút mang giá trị lớn nhất của cây nhị phân tìm kiếm tạo ở câu a.
- (1.0đ) Viết hàm kiểm tra cây có địa chỉ nút gốc là **goc** có phải là cây nhị phân tìm kiếm không? (trả về 0: cây nhị phân tìm kiếm, trả về 1: không là cây nhị phân tìm kiếm).

▪ Nộp bài:

- Chụp ảnh các trang bài làm, tên các ảnh tương thích với thứ tự các trang. Ví dụ: Page1, Page2,...
- Lưu các ảnh trang bài làm vào 1 thư mục có tên: TênSV_Mã số SV, ví dụ: Tuan_62123456, nén lại và nộp lên trang Elearning.

TRƯỜNG ĐẠI HỌC NHA TRANG KHOA CÔNG NGHỆ THÔNG TIN Người ra đề  Nguyễn Đức Thuận	ĐỀ THI KẾT THÚC MÔN HỌC MÔN: CẤU TRÚC DỮ LIỆU & GT Thời gian: 60 phút Lớp: 62 CNTT-4 <i>Không sử dụng tài liệu</i> ĐỀ II Ngày thi: 11 /01/2022	DUYỆT ĐỀ  Nguyễn Đức Thuận
---	---	---

- **Đề này dành cho các sinh viên ký số cuối trong Mã số sinh viên là số chẵn.**

Câu I: (2.0đ) Đánh giá độ phức tạp của đoạn chương trình sau:

```
int HK(int n) {
    for (int i=1; i<=n; i++)
    { int j=1;
      while (j<=n)
        {j= j*2; printf("*");}
    }
```

Câu II: (3.0đ)

- a. (1.5đ) Xây dựng cây nhị phân tìm kiếm lần lượt từ các giá trị sau, trong quá trình xây dựng tại nút thứ i cây không phải là cây AVL thì cân chỉnh lại cây:

15 12 14 25 20 16

- b. (1.5đ) Chuyển đổi biểu thức trung tố sau sang biểu thức hậu tố:

$(9/5 - 8*(2/3*8-3))/7*3$

Câu III: (5.0đ)

Viết chương trình thực hiện các công việc sau:

- (0.5đ) Nhập vào các số nguyên từ bàn phím. Kết thúc khi gõ Enter. Các số nguyên nhập vào được tổ chức lưu trữ thành một danh sách liên kết đơn.
- (0.5đ) In ra các số vừa nhập
- (1.0đ) Chèn nút mang giá trị X đứng trước nút mang giá trị Y (nếu có).
- (1.0đ) Sắp xếp các nút theo giá trị tăng dần.
- (1.0đ) Giả sử danh sách đã được sắp xếp tăng dần, xóa các nút trùng nhau của danh sách, các nút có giá trị trùng nhau chỉ giữ lại 1 nút.
- (1.0đ) Đảo ngược danh sách không tạo ra danh sách mới.

Nộp bài:

- Chụp ảnh các trang bài làm, tên các ảnh tương thích với thứ tự các trang. Ví dụ: Page1, Page2,...
- Lưu các ảnh trang bài làm vào 1 thư mục có tên: TênSV_Mã số SV, ví dụ: Tuan_62123456, nén lại và nộp lên trang Elearning.

ĐÁP ÁN ĐỀ NGHỊ

ĐỀ I

Câu I:

Đánh giá độ phức tạp của đoạn chương trình sau:

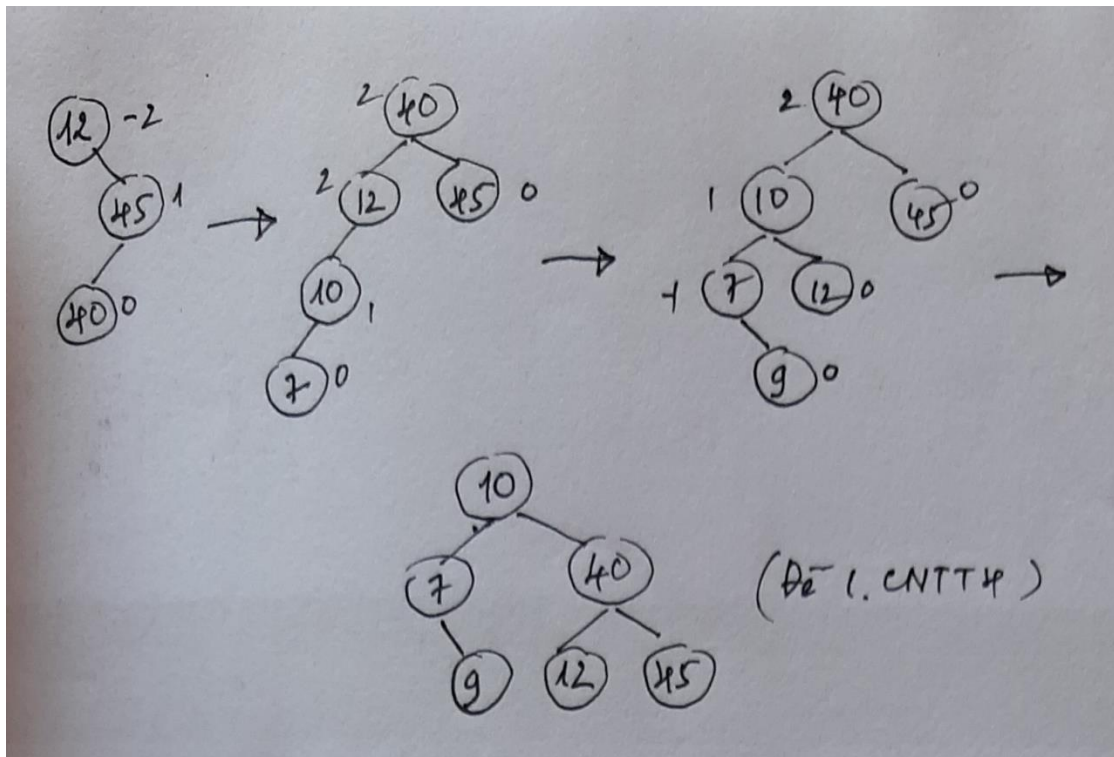
```
int PT (int n) {  
    if (n==0) return 0;  
    for (int i=1; i <= n; i++)  
        for (int j=1; j<=n; j++)  
            { printf("*");  
              break; } }
```

$T(n) = O(n)$; Vòng lặp `for (int j=1; j<=n; j++)`
`{ printf("*");`
`break; }`

chỉ thực hiện 1 lần.

Câu II:

a.



b.

1	2	3	4	5	6	7	8	9	10	11	12
15	20	60	45	30	42	78	10	25	50	17	62
15	20	17	45	30	42	78	10	25	50	60	62
15	20	17	25	30	42	78	10	45	50	60	62
15	20	17	25	30	10	78	42	45	50	60	62
15	10	17	25	30	20	45	42	78	50	60	62
10	15	17				42	45	50	78	60	62
	15	17		20	30				48	60	62

Câu III:

```
#include <stdio.h>
//-----
struct nut {
    int info;
    nut *left,*right;
};

typedef nut Node;
//-----
Node *goc,*goc1;

void khoitao(Node *&goc)
{ goc=NULL;
}
//-----
void mocnut(Node *&goc, int X) // moc 1 nut gtri X vao cay
{ if (goc==NULL)
    { goc=new Node;
      goc->info=X;
      goc->left=NULL;
      goc->right=NULL;
    }
  else
    if (goc->info>= X)
        mocnut(goc->left,X);
    else
        mocnut(goc->right,X);
}
//-----
// a.
void taocay(Node *&goc)
{ int tam;
  do{
    printf("\n Nhap 1 so nguyen,0:dung ");
```

```

        scanf("%d",&tam);
        if (tam!=0)
            mocnut(goc,tam);
    }
    while (tam!=0);
}
//b. -----
void LNR(Node *goc)
{ if (goc!=NULL)
    { LNR(goc->left);
      printf("%3d",goc->info);
      LNR(goc->right);
    }
}
//c. -----

int tonglachan(Node *goc)
{ if (goc==NULL) return 0;
  else
    if (goc->left==NULL && goc->right==NULL && goc->info%2==0)
      return goc->info;
    else
      return tonglachan(goc->left)+tonglachan(goc->right);
}
//d. -----
void chepcay(Node *goc,Node *&goc1)
{ if (goc==NULL) goc1=NULL;
  else
    { goc1=new Node;
      goc1->info=goc->info;
      chepcay(goc->left,goc1->left);
      chepcay(goc->right,goc1->right);
    }
}
//e. -----
int lonnhat(Node *goc)
{ if (goc==NULL) return -1111;
  else
    if (goc->right==NULL) return goc->info;
    else return lonnhat(goc->right);
}
//f. -----
int ktra(Node *goc)
{ if (goc==NULL) return 0;
  else
    if ((goc->left==NULL)&&(goc->right==NULL))

```

```

        return 0;
    else
        if (((goc->left!=NULL)&& (goc->info<goc->left->info))
            || ((goc->right!=NULL)&& (goc->info>goc->right->info)))return 1;
        else
            if ((ktra(goc->left)+ktra(goc->right))==0) return 0;
            else
                return 1;

    }
    //-----
int main()
{
    khoitao(goc);
    khoitao(goc1);
    taocay(goc);
    printf("\n Duyệt cay LNR \n");
    LNR(goc);
    printf("\n Tong la chan %d \n",tonglachan(goc));
    printf("\n Gia tri lon nhat %d \n",lonnhat(goc));
    chepcay(goc,goc1);
    printf("\n Cay goc: \n");
    LNR(goc);
    printf("\n Cay sao chep: \n");
    LNR(goc1);

    if (ktra(goc)==0) printf("\n Cay nptk \n");
    else printf("\n Cay khong la cay nptk \n");
    return 0;
}

```

ĐỀ II

Câu I:

Đánh giá độ phức tạp của đoạn chương trình sau:

```

int HK(int n) {
    for (int i=1; i<=n; i++)
    {
        int j=1;
        while (j<=n)
        {
            j=j*2; printf("*");
        }
    }
}

```

Vòng lặp:

```

int j=1;
while (j<=n)
{
    j=j*2; printf("*");
}

```

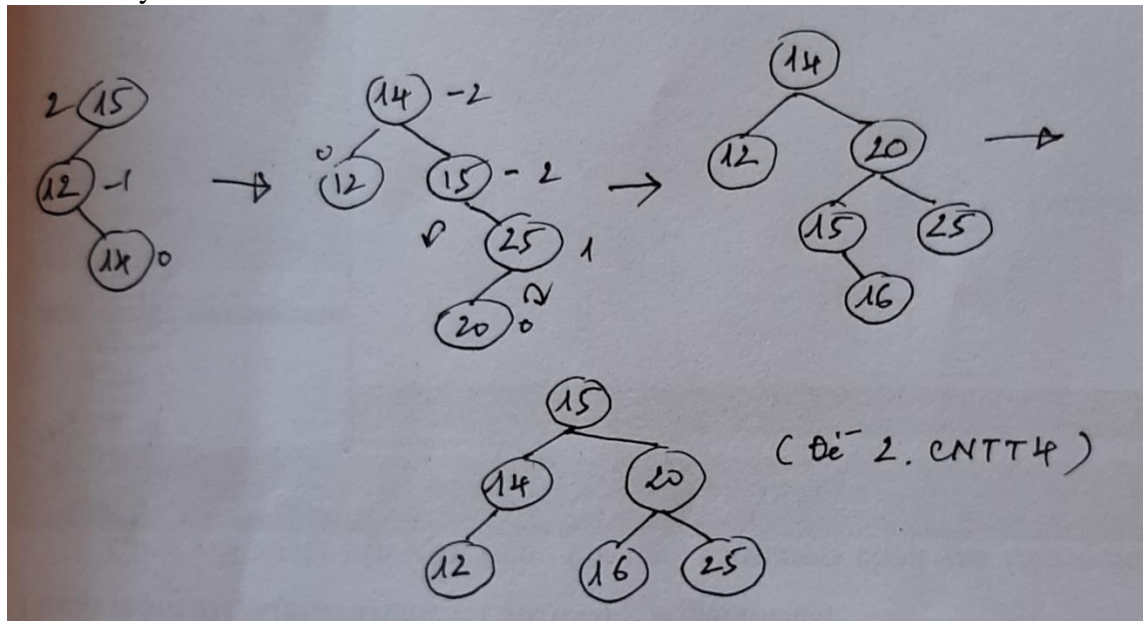
- Số lần lặp: $k = \log_2 n$.

Vòng lặp: for (int i=1; i<=n; i++) lặp n lần.

$$T(n) = O(n \log_2 n).$$

Câu II:

a. Cây AVL



b. Biểu thức hậu tố: 9 5 / 8 2 3 / 8 * 3 - * - 7 / 3 *

Câu III:

```
#include <stdio.h>
//-----
struct nut {
    int gtri;
    nut *tiep;
};
typedef nut Node;
//-----
Node *dau, *daumoi;
//-----
void khoitao(Node *&dau)
{
    dau = NULL;
}
//-----
void nhapds(Node *&dau)
{
    int tam;
    Node *p, *q;
    q = NULL;
    do {
        printf("\n Nhập vào 1 số nguyên, 0: dừng :");
        scanf("%d", &tam);
```

```

        if (tam!=0)
        { // Tao nut moi
            p= new Node;
            p->gtri=tam;
            p->tiep=NULL;
            // Moc vao danh sach
            if (dau==NULL)
                dau=p;
            else
                q->tiep=p;
                q=p;
            }
        }
    while(tam!=0);
}
//-----
//--- Duyet de qui
void duyetdq(Node *dau)
{ if (dau!=NULL)
    { printf("%3d",dau->gtri);
      duyetdq(dau->tiep);
    }
}
//-----
// Bo sung 1 nut mang gia tri X vao dau ds
void chendau(Node *&dau,int X)
{ Node *p;
  p=new Node;
  p->gtri=X;
  p->tiep=dau;
  dau=p;
}
//-----
void chentruoc(Node *&dau,int X, int Y)
{ Node *p,*q,*t;
  // t: tro den nut Y; q: tro den nut truoc Y
  // p: dia chi nut X
  q=NULL;
  t=dau;
  // Tim nut mang gia tri
  while((t!=NULL)&&(t->gtri!=Y))
  { q=t;
    t=t->tiep;}
    if (t!=NULL) // co nut mang gia tri Y
        if (t==dau)
            chendau(dau,X);
}

```



```

        else
        { // Tao nut moi chua X
          p=new Node;
          p->gtri=X;
          // Moc nut
          q->tiep=p;
          p->tiep=t;
        }
      }
//-----
void sapxep(Node *&dau)
{ Node *p,*q;
  int tam;
  if (dau!=NULL)
  {   p=dau;
      while (p->tiep!=NULL)
      { q=p->tiep;
        while (q!=NULL)
        { if (q->gtri<p->gtri)
          { tam = p->gtri;
            p->gtri=q->gtri;
            q->gtri=tam;
          }
          q=q->tiep;
        }
        p=p->tiep;
      }
  }
}
//-----
void xoatrung(Node *&dau)
{ Node *p,*q,*t;
  p=dau;
  while (p->tiep!=NULL)
  { q=p->tiep;
    if (p->gtri==q->gtri)
    { p->tiep=q->tiep;
      delete q;
    }
    else
    p=p->tiep;
  }
}
//-----
void daonguocktds(Node *&dau)
{ Node *p,*q;
  q=NULL; p=NULL;

```

```

        while (dau!=NULL)
        { p=dau;
          dau=dau->tiep;
            p->tiep=q;
            q=p;
          }
        dau=p;
    }
//-----
int main()
{ khoitao(dau);
  nhapds(dau); //a.
  duyetdq(dau); //b.
  chentruoc(dau,5,7); //c.
  printf("\n Chen truoc \n");
  duyetdq(dau);
  printf("\n Chen sap xep \n");
  sapxep(dau); //d.
  duyetdq(dau);
  printf("\n Xoa trung \n");
  xoatrung(dau); //e.
  duyetdq(dau);
  printf("\n Dao nguoc \n");
  daonguocktds(dau);
  duyetdq(dau);
  return 0;
}

```

