Solutii la examenul de Structuri de Date
din 15 ianuarie 2011, ora 10:30, sala 2104, an 3 IDD

1. Scrieti procedura pentru afisarea rezultatelor obtinute in cadrul proiectului.

Solutii:

Daca in proiect este folosita dominant lista simpla, procedura este:

a) traversare lista simpla

```
void traversare(nod *cap)
{
	nod *p;
	float val=0;
	printf("\nCod Cantitate Pret Valoare");
	p=cap;
	while(p)
	{
	printf("\n%5.2f %5.2f %5.2f %5.2f",p->cod, p->cant,p->pret,
	(p->cant)*(p->pret));
	val+=(p->cant)*(p->pret);
	p=p->next;
	}
	printf("\nValoare totala=%5.2f",val);
}
```

unde structura nodului listei simple este:

```
struct nod
{
	  float cod, cant, pret;
	  struct nod* next;
};
```

Daca in proiect este folosita dominant lista dubla, procedura este:

b) traversare lista dubla

```
void traversare(nod *cap)
{
	nod *p;
	float val=0;
	printf("\nCod Cantitate Pret Valoare");
	p=cap;
	while(p)
	{
	printf("\n%5.2f %5.2f %5.2f %5.2f",p->cod, p->cant,p->pret,
	(p->cant)*(p->pret));
	val+=(p->cant)*(p->pret);
	p=p->next;
	}
	printf("\nValoare totala=%5.2f",val);
}
```

unde structura nodului listei duble este:

```
struct nod
{
        float cod, cant, pret;
        struct nod *next, *prev;
};
```

Daca in proiect este folosit dominant arborele binar, procedura este:

c) traversare arbore binar

```
void afisInOrdine(bynarytreenode* root)
{
      if (root)
      {
            afisInOrdine(root->left);
            printf("\n ISBN = %d, price = %5.2f", root->inf->ISBN,
            root->inf->price);
            afisInOrdine(root->right);
      }
}
```

unde structura articolului si a nodului arborelui sunt:

```
struct book
{
      int ISBN;
      float price;
};

struct bynarytreenode
{
      bynarytreenode* left;
      book* inf;
      bynarytreenode* right;
};
```

---

2. Scrieti procedura de stergere a unui element dintr-o matrice rara reprezentata prin trei vectori.

Solutie:

```
MatriceRara stergere(MatriceRara a, int i, int j)
{
      MatriceRara b;
      b.m=b.n=b.nrnenule=0;
      b.col=b.lin=b.val=NULL;
      for(int k=0;k<a.nrnenule;k++)
      if((a.lin[k]==i)&&(a.col[k]==j))
      {
            b.val=(int*)malloc((a.nrnenule-1)*sizeof(int));
            b.col=(int*)malloc((a.nrnenule-1)*sizeof(int));
            b.lin=(int*)malloc((a.nrnenule-1)*sizeof(int));
            b.m=a.m;
            b.n=a.n;
            b.nrnenule=a.nrnenule-1;
            for(int k1=0;k1<k;k1++)
            {
```

```
                b.lin[k1]=a.lin[k1];
                b.col[k1]=a.col[k1];
                b.val[k1]=a.val[k1];
            }
            for(int k2=k;k2<a.nrnenule-1;k2++)
            {
                b.lin[k2]=a.lin[k2+1];
                b.col[k2]=a.col[k2+1];
                b.val[k2]=a.val[k2+1];
            }
        }
return b;
}
```

unde structura matrice rara, reprezentata prin trei vectori este:

```
struct MatriceRara
{
      int *val;
      int *lin;
      int *col;
      int m;
      int n;
      int nrnenule;
};
```

---

3. Scrieti procedura pentru concatenarea a cinci liste simple.

Solutie:

Concatenarea a doua liste:

```
nod* concatenare(nod *cap1, nod *cap2)
{
      nod *cap3;
      if ((cap1==NULL)&&(cap2==NULL)) cap3=NULL;
      else if ((cap1!=NULL)&&(cap2==NULL)) cap3=cap1;
      else if ((cap1==NULL)&&(cap2!=NULL)) cap3=cap2;
      else
      {
            nod *p;
            for (p=cap1;p->next;p=p->next);
            p->next=cap2;
            cap3=cap1;
      }
      return cap3;
}
```

Apel concatenare doua liste:

```
cap1=concatenare(cap1, cap2);
cap1=concatenare(cap1, cap3);
cap1=concatenare(cap1, cap4);
cap1=concatenare(cap1, cap5);
```

unde structura nodului listei simple este:

```
struct nod
{
float cod,cant,pret;
struct nod* next;
};
```

---

4. Scrieti procedura pentru calculul valorii stocului final al unui material pentru care este dat codul intr-un arbore binar.

Solutie:

Cautare recursiva a materialului cu un anumit cod:

```
bynarytreenode* cauta_recursiv(bynarytreenode* root, int cheie)
{
	if (root)
	{
		if (cheie==root->info.cod) return root;
		else if (cheie<root->info.cod)
		cauta_recursiv(root->left,cheie);
		else cauta_recursiv(root->right,cheie);
	}
	else return NULL;
}
```

Apel cautare recursiva si calcul valoare stoc final, determinata astfel:

val_stoc_final = (stoc_initial + intrari – iesiri)*pret;

```
int cod;
printf("\n Codul de cautat = ");
scanf("%d",&cod);
bynarytreenode* cautat = NULL;
cautat=cauta_recursiv(r,cod);
if (cautat) printf("\n Materialul cu codul %d are stocul initial %d,
intrari %d, iesiri %d, pret %5.2f, valoare stoc final %5.2f",
cautat->info.cod, cautat->info.stoc_initial, cautat->info.intrare,
cautat->info.iesire, cautat->info.pret, (cautat->info.stoc_initial +
cautat->info.intrare - cautat->info.iesire)*cautat->info.pret);
else printf("\n Cod negasit!");
```

unde structura articolului si a nodului arborelui sunt:

```
struct material
{
	int cod;
	int stoc_initial;
	int intrare;
	int iesire;
	float pret;
};

struct bynarytreenode
{
	material info;
	bynarytreenode *left, *right;
```

```
};
```

---

5. Scrieti procedura care permite traversarea unei liste duble in ambele sensuri, la alegere.

Solutie:

```
void afisare(nod *p)
{
	printf("\n%5.2f %5.2f %5.2f %5.2f",p->pr.cod, p->pr.cant,
	p->pr.pret,p->pr.val);
}

void traversare(nod *prim, nod* ultim, int directie)
{
	nod *p;
	printf("\nCod Cantitate Pret Valoare");
	if (directie==0)
	{
		p=prim;
		while(p)
		{
			afisare(p);
			p=p->next;
		}
	}
	else if (directie==1)
	{
		p=ultim;
		while(p)
		{
			afisare(p);
			p=p->prev;
		}
	}
}
```

unde structura articolului si a nodului listei duble sunt:

```
struct produs
{
	float cod,cant,pret,val;
};

struct nod
{
	produs pr;
	struct nod *prev, *next;
};
```

---

6. Scrieti procedura pentru alegerea minimului sau maximului dintre elementele unei structuri de date, la alegere.

Solutie:

Determinare element cu cod minim sau cod maxim din vectorul de produse cu *n* componente:

```
int min_max(produs *p, int n, int vs)
{
	int min, max, result;
	if (vs==0)
	{
		min=p[0].cod;
		for (int i=1;i<n;i++) if (min>p[i].cod) min=p[i].cod;
		result=min;
	}
	else if (vs==1)
	{
		max=p[0].cod;
		for (int i=1;i<n;i++) if (max<p[i].cod) max=p[i].cod;
		result=max;
	}
	else result=0;
return result;
}
```

unde structura articolului este:

```
struct produs
{
	int cod;
	float cantitate;
	float pret;
};
```

---

7. Scrieti procedura care preia informatia utila dintr-un arbore binar si o copiaza intr-o lista simpla.

Solutie:

Inserare element in lista simpla:

```
listnode* inserare(listnode* cap, book *b)
{
	listnode* nou = (listnode*)malloc(sizeof(listnode));
	listnode* temp;
	nou->inf=b;
	nou->next=NULL;
	if (cap==NULL) return nou;
	temp=cap;
	while (temp->next) temp=temp->next;
	temp->next=nou;
	return cap;
}
```

Procedura conversie arbore binar – lista simpla:

```
listnode* cap = NULL;

void bynarytree_to_list(bynarytreenode *root)
{
	if (root)
```

```
        {
                cap=inserare(cap, root->inf);
                bynarytree_to_list(root->left);
                bynarytree_to_list(root->right);
        }
}
```

unde structura articolului, a nodului arborelui si a nodului listei simple sunt:

```
struct book
{
     int ISBN;
     float price;
};

struct bynarytreenode
{
     bynarytreenode* left;
     book *inf;
     bynarytreenode* right;
};

struct listnode
{
     book *inf;
     listnode *next;
};
```

---

8. Indicati trei argumente care sa justifice alegerea structurii de date dinamice din proiect.

Solutie:

Primul argument este legat de *operatiile* care se realizeaza cu structura de date dinamica. Cel de-al doilea argument se refera la *volumul de prelucrari*. Al treilea argument este legat de existenta *bibliotecilor de proceduri* care folosesc respectiva structura de date dinamica.

---

9. Aratati care a fost eroarea de compilare cea mai frecvent intalnita cand ati scris programele din proiect, specificati cauze si aratati cum ati efectuat corectiile.

Solutie:

Se prezinta una dintre *erorile de sintaxa* care presupun corectii pe textul sursa din punct de vedere al limbajului de programare. Se descrie una dintre *erorile de logica a programarii* care necesita corectii severe in cadrul expresiilor.

---

10. Scrieti o procedura de creare a unei structuri de date existenta in proiectul elaborat.

Solutii:

Daca structura de date este arborele binar, procedura este:
a) creare nod arbore binar

```
bynarytreenode* createnode(student stud, bynarytreenode *l, bynarytreenode
*r)
{
	bynarytreenode *temp;
	temp=(bynarytreenode*)malloc(sizeof(bynarytreenode));
	(*temp).info.cod=stud.cod;
	temp->info.nume=(char*)malloc((strlen(stud.nume)+1)*sizeof(char));
	strcpy(temp->info.nume,stud.nume);
	temp->info.varsta=stud.varsta;
	temp->left=l;
	temp->right=r;
	return temp;
}
```

unde structura articolului si a nodului arborelui sunt:

```
struct student
{
	int cod;
	char* nume;
	int varsta;
};

struct bynarytreenode
{
	student info;
	bynarytreenode *left, *right;
};
```

Daca structura de date este lista dubla cu *n* noduri, procedura este:

b) creare lista dubla cu *n* noduri

```
nod* crearelista(nod *cap, int n)
{
 nod *p, *q;
 float cod1,cant1,pret1;
 citire(&cod1,&cant1,&pret1);
 cap=(nod*)malloc(sizeof(nod));
 //
 //fiecare pune atribuirile care se potrivesc problemei rezolvate
 //
 cap->pr.cod=cod1;
 cap->pr.cant=cant1;
 cap->pr.pret=pret1;
 cap->prev=NULL;
 cap->next=NULL;
 p=cap;
 for (int i=1;i<n;i++)
 {
	citire(&cod1,&cant1,&pret1);
	q=(nod*)malloc(sizeof(nod));
```

```
        q->pr.cod=cod1;
        q->pr.cant=cant1;
        q->pr.pret=pret1;
        q->next=NULL;
        q->prev=p;
        p->next=q;
        p=q;
}
return cap;
}
```

unde structura articolului si a nodului listei sunt:

```
struct produs
{
      float cod, cant, pret;
};

struct nod
{
      produs pr;
      struct nod *prev, *next;
};
```
