

# J0186Fonaments d'Informatàtica (103806) Curs 2020-

2021  
Examen Parcial (23 de Novembre de 2020)

Nom estudiant: JOSE ELSTON MARTINEZ Grup: NIU: 4172900

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més

de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherència entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartrons tenen un total de 25 números. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartró. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

## Exercici 1 (1 punt)

Fer un procediment anomenat CrearCartró per a generar un cartró. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 31 al 45,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartró) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- int GenerarNombre (int min, int max, int a[], int dim)  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- void Ordena (int a[], int dim)  
Ordena l'array a de dimensió dim de menor a major.

$$35 \times 15 = 525$$
$$75 + 25 = \underline{100}$$

## Exercici 2 (1 punt)

Fer la funció anomenada ArraysIguals per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retorna true en el cas que siguin iguals i false en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$325 + 150 = 475$$

$$\text{RESULTAT} : 35$$

## Exercici 3 (1 punt)

Fer un procediment anomenat ImprimirBolesSortides per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

NOTA: Recordeu que els números de les boles van de 1 a 75.

#### Exercici 4 (1 punt)

Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contindrà informació dels nombres que han sortit. En l'array bombo si el valor d'una determinada posició (index) és un 1, vindrà dir que la bola corresponent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

```
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Assignar cartrons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
Inicialitza el vector a de dimensió dim al valor v
```

```
int GenerarNombre (int min, int max, int a[], int dim)
Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim
```

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

1. Declarar (on corresponguï) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on corresponguï) els arrays necessaris per representar els cartrons de dos jugadors anomenats cartroPlayer1 i cartroPlayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment MenuPrincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartrons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CreateCartro per crear els cartrons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartrons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartrons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menu principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els noms de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge general pel procediment ImprimirComentari.
7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa..."
8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.

#### Exercici 6 (2.5 punts)

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenuPrincipal()
{
    cout << " --- Menú Principal --- " << endl;
    cout << " 1.- Asignar cartrons " << endl;
    cout << " 2.- Jugar " << endl;
    cout << " 3.- Marcador " << endl;
    cout << " 4.- Sortir " << endl;
}
```



**Exercici 7 (0.5 punts)**

Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `Non_Jugador` (cadena de 50 caràcters). Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeixi l'identificador del cartró i el nom del jugador separant-los amb el caràcter ‘,’.

**Exercici 9 (1 punt)**

Escriure la funció `Elevat` que retorni el resultat d'elevant un nombre al quadrat. La funció tindrà 2 paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'intervall [-46.340,46.340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada `Es_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

```

int Vofar (int min, int max)
int rot;
cout << ... ;
cin >> rot;
while (rot < min || rot > max)
    cout << ... ;
    cout << ... ;
    cin >> rot;
return rot;

```

Void InicializarArray (int V[], int dim, int valor)

```

int i;
for (i=0; i<dim; i++)
    V[i] = valor;

```

int NOZeroArray (int V[], int dim)

```

int i=0;
bool trobat=false;

```

While (i < dim) && (!trobat)

```

if (V[i]==0)
    trobat=true;
else
    i++;

```

if (trobat==true)

return 0;

else

return 1;

return ind;

ind = i;

min = V[3];

$\text{if } (\text{V}[3] < \text{min}) \& (\text{V}[3] \neq 0)$

(++) i; dim; i = 1) - 10

min = V[3];

ind = 0;

if (!min, ind);

int res = (int) V[3], int dim;

res +=

$\text{if } (\text{V}[3] == y)$

(++) i; i < dim; i++)

for (j = 0; j < dim; j++)

SWHILE ( $j < \text{OutputSize}$ )

res = 0;

Results[j] = res;

++res

$\text{if } (\text{V}[3] == y)$

for (i = 0; i < dim; i++)

do

$y = 0$

res

$y =$



void ESCRUFAN (int V[3], int dim, int Results[])

# Ronaments d'Informàtica (103806)

Curs 2020-2021

Examen Parcial (23 de Novembre de 2020)

Grup: 5864644

Nom estudiant: Miquel Solomons

NIU:

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bumbo. Els cartells tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bumbo i es guanya un premi quan han sortit tots els nombres del nostre cartell. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

## Exercici 1 (1 punt)

Fer un procediment anomenat CrearCartell per a generar un cartell. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 31 al 45,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartell) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- int GenerarNombre (int min, int max, int a[], int dim)
- Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- void Ordena (int a[], int dim)
- Ordena l'array a de dimensió dim de menor a major.

$$15 + 15 = 30$$

$$35 - 10 = \underline{25}$$

## Exercici 2 (1 punt)

Fer la funció anomenada ArraysIguals per saber si dos arrays tenen tots els valors idèntics.

La funció rebrà dos arrays i la seva dimensió. Retorna true en el cas que siguin iguals i false en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$20 \times 4 = 80$$

$$30 + 20 = 50$$

Resultat: 50

## Exercici 3 (1 punt)

Fer un procediment anomenat ImprimirBolesSortides per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

NOTA: Recordeu que els números de les boles van de 1 a 75.

#### Exercici 4 (1 punt)

Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (ombo) que contindrà informació dels nombres que han sortit. En l'array bombo si el valor d'una determinada posició (índex) és un 1, voldrà dir que la bola correspondent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

#### Exercici 6 (2.5 punts)

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void Menuprincipal()
{
    cout << "--- Menu Principal ---" << endl;
    cout << "1.- Asignar cartrons" << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Assignar cartrons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
{
    int GenerarNombre(int min, int max, int a[], int dim)
}
```

Inicialitza el vector a de dimensió dim al valor v

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main()) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartons de dos jugadors anomenats cartroplayer1 i cartroplayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment Menuprincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CrearCartro per crear els cartons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartons són diferents utilitzant la funció AreigsIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menu principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els nombres de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa....".
8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0,5 punts)**

Declarar un nou tipus de dades, TCartro, com un registre amb els camps: Ident\_Carro (valor enter), Numeros (array de 25 enters) i Nom\_Jugador (cadena de 50 caràcters). Fer un procediment anomenat Imprimir que rebi com a paràmetre un registre del tipus TCartro i imprimeix l'identificador del carro i el nom del jugador separant-los amb el caràcter ‘,’.

**Exercici 9 (1 punt)**

Escriure la funció Elevar que retorni el resultat d'elevat un nombre al quadrat. La funció tindrà 2 paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realitzar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

**NOTA:** El valor de la base ha d'estar inclos en l'intervall [-46,340,46,340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada Es\_MatPositiva que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

# J0186Fonaments d'Informàtica (103806) Curs 2020-

2021

Examen Parcial (23 de Novembre de 2020)

Nom estudiant: AMBER VÉTETEL

Grup: NIU: 3454743

**Exercici 2 (1 punt)**  
Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics.  
La funció rebrà dos arrays i la seva dimensió. Retornarà `True` en el cas que siguin iguals i `False` en cas contrari.

**NOTA:** Podeu suposar que els dos arrays tenen la mateixa dimensió.

**Important:** Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)  
Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherència entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartons tenen un total de 25 números. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els números del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

## Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartó` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 11 al 15,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

**UTILITZEU (no implementar) els següents procediments o funcions:**

- `int GeneraNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre `min` i `max`, i revisa que no estigui a l'array a de dimensió `dim`.
- `void Ordena (int a[], int dim)`  
Ordena l'array a de dimensió `dim` de menor a major.

$$85 \times 25 = 2125$$
$$100 - 35 = \boxed{65}$$

## Exercici 3 (1 punt)

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contendrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`index`) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

**NOTA.** Recordeu que els números de les boles van de 1 a 75.

$$72 + 72 = 144$$
$$2 \times 70 = 140$$

#### Exercici 4 (1 punt)

Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contindrà informació dels nombres que han sortit. En l'array bombo si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

#### Exercici 6 (2.5 punts)

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenuPrincipal()
{
    cout << "---- Menú Principal ----" << endl;
    cout << "1.- Asignar cartrons" << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Assignar cartrons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
```

Inicialitzar el vector a de dimensió dim al valor v

```
int GenerarNombre(int min, int max, int a[], int dim)
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

1. Declarar (o correpondrà) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (o correpondrà) els arrays necessaris per representar els cartrons de dos jugadors cada jugador. En el cas que la diferència sigui:
  - menor o igual que 1, escriure el missatge "Està molt igualat".
  - entre 2 i 5, el missatge hauria de ser "La remuntada és possible".
  - Major o igual que 6, el missatge haurà de ser "Avui és el teu dia de sort".
- NOTA: podeu utilitzar la funció abs() per obtenir el valor absolut d'un nombre.
3. Utilitzar el procediment MenuPrincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartrons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CrearCartró per crear els cartrons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartrons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. Shaurà de comprovar que s'han generat els cartrons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els nombres de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa....".
8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0.5 punts)**

Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `Nom_Jugador` (cadena de 50 caràcters).

Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeixi l'identificador del cartró i el nom del jugador separant-los amb el caràcter ‘`,`

**Exercici 9 (1 punt)**

Escriure la funció `Elevar` que retorna el resultat d'elevat. La funció tindrà dos paràmetres de tipus `int`. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realitzar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclòs en l'intervall [-46,340,46,340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada `Es_MatPositiva` que rebi com a paràmetres: una matríu d'enters de 4 columnes, el número de fileres de la matríu, i el número de columnes de la matríu, i retorna 1 si tots els valors de la matríu són positius o zero, i un 0 en cas contrari.

### Ex. 2

```
int Voter (int min, int max)
```

```
{ int vot;
```

```
cout << "entre valor min " << min << " e max " << "
```

```
<< n >> vot;
```

```
while ((vot < min) || (vot > max))
```

```
{
```

```
cout << "Error";
```

```
cout << "entre valor entre ...";
```

```
<< n >> vot;
```

```
}
```

```
return vot;
```

### Ex. 5

```
void Escritorio (int votos[], int dm, int resultado)
```

```
for (int i = 0; i < dm; i++)
```

```
resultado[votos[i]]++;
```

```
}
```

### Ex. 6

```
int NoZeroArray (int a[], int dm)
```

```
{
```

```
bool trobat = false;
```

```
int i = 0;
```

```
while ((i < dm) && (!trobat))
```

```
{
```

```
if (a[i] == 0)
```

```
trobat = true;
```

```
else
```

```
i++;
```

```
}
```

```
return !trobat;
```

```
int indexMinNoZeroArray (int a[], int dm)
```

```
{ int valmin = a[0];
```

```
int posmin = 0;
```

```
for (int i = 1; i < dm; i++)
```

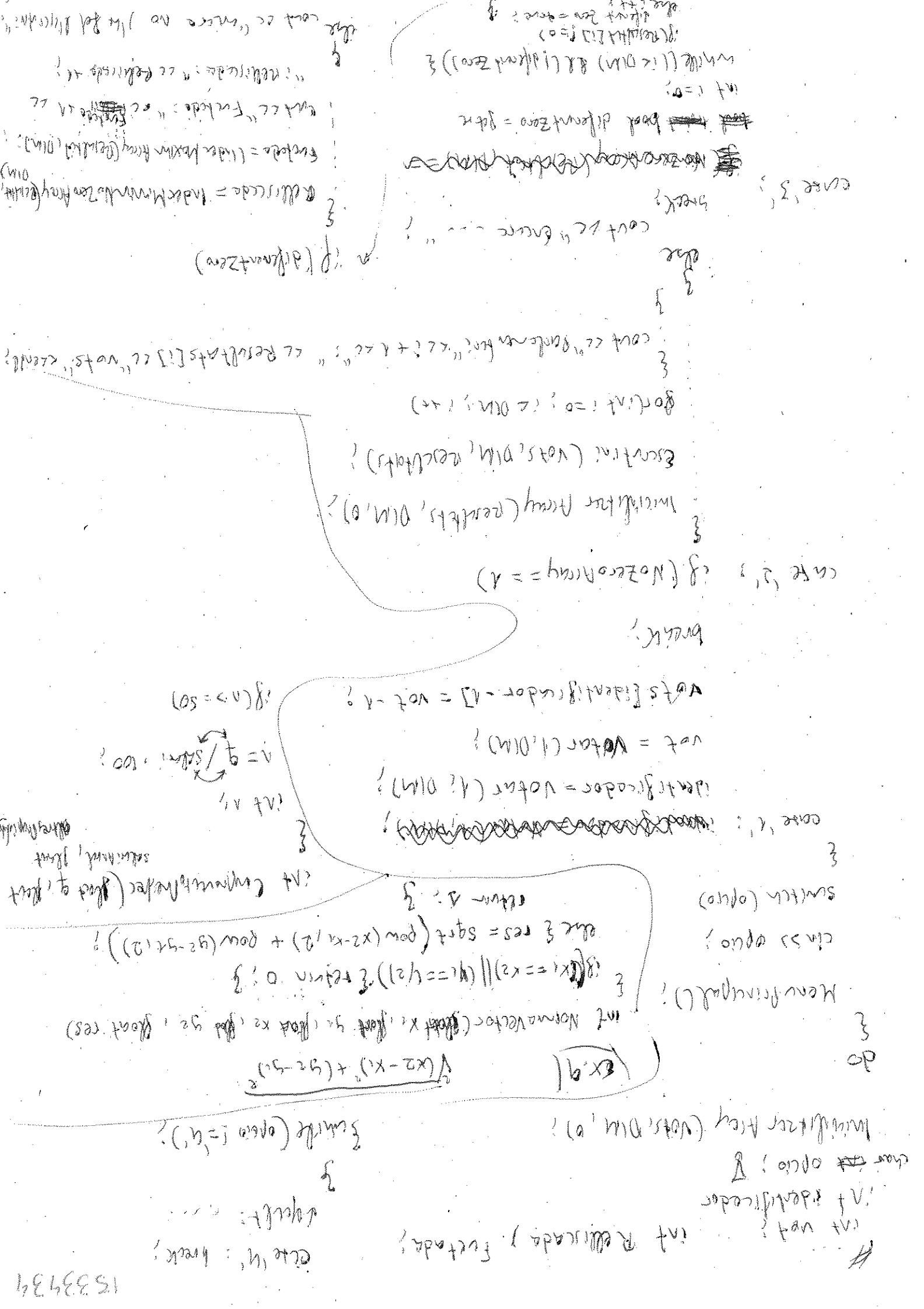
```
if (a[i] < valmin) && (a[i] != 0)
```

```
valmin = a[i];
```

```
posmin = i;
```

```
for (int i = 0; i < dm; i++)
```

```
return posmin;
```



## Periodistes del parlament

Fuetada  $\rightarrow$  millor { intervenció d'en parlamentari.  
 Pellscaada  $\rightarrow$  pitjar

Diputats  $\rightarrow$  135

Ordenats alfabèticament  $\rightarrow$  assignant valors de l's -135.

A peu

1	2
---	---

0

2

135
-----

136

Todos dolen dolor

Alfabèticament ordenats.

Alfabet 23.

1	:	:	:	:	:
---	---	---	---	---	---

0

135
-----

134.

KD

1	2	3	4	5
---	---	---	---	---

0 1 2 3 4.

$\rightarrow$  Valor  $\rightarrow$

1	1	1	1	1	1
---	---	---	---	---	---

0

0
---

Per ( $i = 0$ ;  $i < \text{dimV}$ ;  $i++$ )

1	1	1	1	1	1
---	---	---	---	---	---

 $\rightarrow$  0

$i = 0$ .

1	1	1	1	1
---	---	---	---	---

 $\rightarrow$  0:

1	1	1	1	0	1	1
---	---	---	---	---	---	---

1	1	1
---	---	---

 $\rightarrow$  1.

else return 0;

else  $P_{\text{unblock}} = 1$

else if ( $(\text{selecr} < (\text{Prestle} \times 10) / 100)$ )  $\ll$   $(\text{order} / 2) > (\text{Prestle})$ )

$\text{order} < \text{Prestle}$

$P_{\text{unblock}} = P_1 + P_2$

else if ( $(\text{selecr} < (\text{Prestle} \times 25) / 100)$ )  $\ll$   $(\text{order} / 2) > (\text{Prestle})$ )

$\text{order} < \text{Prestle}$

$P_{\text{unblock}} = P_1 + P_2$

} }  $(\text{selecr} < (\text{Prestle} \times 2) > (\text{Prestle}))$

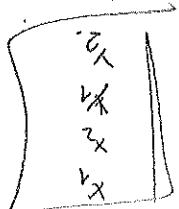
$$1.92 - 1.01 \Rightarrow 5 \%$$

$$(1.05 - 50\%) \Rightarrow 5 \%$$

$$50\% \Rightarrow 5 \%$$

$(x_1, y_1)$

$(x_2, y_2)$



0	1	2	3	4	5	6
1	2	3	4	5	6	7

# Examen Parcial (23 de Novembre de 2020)

Nom estudiant: James Davis Trudeau

Curs 2020-2021

Grup: 5089484

NIU: 5089484

Exercici 1 (1 punt)  
Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `true` en el cas que siguin iguals i `false` en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$145 - 20 = 95$$

$$95 + 5 = 100$$

Resultat: 100

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de 11 al 75 dins del bombo. Els cartrons tenen un total de 25 números. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartró. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cada soci.

Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartro` per a generar un cartró. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 11 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 31 al 45,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartró) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- `int GenerarNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordena l'array a de dimensió dim de menor a major.

$$4 \times 10 = \underline{40}$$

$$40 - 9 = \underline{31}$$

Exercici 3 (1 punt)

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`index`) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

NOTA. Recordeu que els números de les boles van de 1 a 75.

#### Exercici 4 (1 punt)

Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contingrà informació dels nombres que han sortit. En l'array bombo si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola correspondent ha sortit; si el és un 0, voldrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

#### Exercici 6 (2,5 punts)

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenúPrincipal()
{
    cout << "---- Menú Principal ----" << endl;
    cout << "1.- Asignar cartrons" << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Asignar cartrons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
{
    for (int i = 0; i < dim; i++)
        a[i] = v;
}
```

Inicialitza el vector a de dimensió dim al valor v

```
int GenerarNombre(int min, int max, int a[], int dim)
{
    return rand() % (max - min + 1) + min;
}
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main()) que segueixi els següents passos:

1. Declarar (on corresponguï) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on corresponguï) els arrays necessaris per representar els cartrons de dos jugadors anomenats cartróJugador1 i cartróJugador2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment MenúPrincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CrearCartró per crear els cartons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els nombres de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa...".
8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0.5 punts)**  
Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `Nom_Jugador` (cadena de 50 caràcters).

Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeixi l'identificador del cartó i el nom del jugador separant-los amb el caràcter ‘`,`’.

**Exercici 9 (1 punt)**  
Escriure la funció `Elevar` que retorna el resultat de elevar un nombre al quadrat. La funció tindrà~ paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si s'ha pogut realizar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'intervall [-46.340,46.340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada `Es_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

**Examen Parcial (23 de Novembre de 2020)** Grup:

**Nom estudiant:** FRANCINE FLORES NIU: 1887624

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més

de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartrons tenen un total de 25 números. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

#### Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartó` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 11 al 15,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- `int GenerarNúmero(int min, int max, int a[], int dim)`
- `void Ordena(int a[], int dim)`

NOTA. Recordeu que els números de les boles van de 1 a 75.  
Ordene l'array a dimensió dim de menor a major.

$$10 + 10 = 20$$

$$35 + 10 = \underline{45}$$

#### Exercici 2 (1 punt)

Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `true` en el cas que siguin iguals i `false` en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$30 + 40 = 70$$

$$\text{RESULTAT} = 75$$

#### Exercici 4 (1 punt)

Fer una funció anomenada Escrutini, per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contindrà informació dels nombres que han sortit. En l'array bombó si el valor d'una determinada posició (índex) és un 1, vindrà dir que la bola corresponent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

#### Exercici 6 (2,5 punts)

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void InicialitzarArray(int a[], int dim, int v)
{
    cout << "---- Menu Principal ----" << endl;
    cout << "1.- Assignar cartrons" << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Assignar cartrons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
{
    int GenerarNombre (int min, int max, int a[], int dim)
    {
        // Implementació de GenerarNombre
    }
}
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartrons de dos jugadors anomenats cartroPlayer1 i cartroPlayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment menuPrincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartrons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CrearCartro per crear els cartrons del jugador 1 i del jugador 2.
    - menor o igual que 1, escriure el missatge "Està molt igualat".
    - entre 2 i 5, el missatge hauria de ser "La remuntada és possible".
    - Major o igual que 6, el missatge haurà de ser "Avui és el teu dia de sort".
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartrons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els nombres de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa...".
8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0,5 punts)**

Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `Nom_Jugador` (cadena de 50 caràcters).  
Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeixi l'identificador del cartó i el nom del jugador separant-los amb el caràcter `:`.

**Exercici 9 (1 punt)**

Escrivre la funció `Elevar` que retorna el resultat d'elevat un nombre al quadrat. La funció tindrà dos paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si s'ha pogut realizar l'operació i un 0 en cas contrari.

**NOTA:** El valor de la base ha d'estar inclos en l'intervall [-46,340,46,340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada `Es_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

```

Menu →
100 int voto;
OPCIÓN 2 = do {
    cout << "Introduce el voto" << endl;
    cin >> Voto;
    { while ((Voto > min) && (Voto < max)) →
        votar (int min, int max);
        for (char i = 0; i < pm; i++) {
            VOTOS[i];
        }
    }
}
OPCIÓN 2 = int VOTOS[pm];
OPCIÓN 2 = Array [E, Bm]
No cero Array (nm);
if (Votos < N_Votos)
{
    Inicializar Array (nm);
    cout << VOTOS[i];
}
else
{
    cout
}
OPCIÓN 3 =
do
{
    IndiceMínimo No cero Array (Array [E, Bm]);
    int Indice Máximo Array / J;
}
while (escrutinio == True)
if (escrutinio == False)
{
    cout << mensajes
}
}
3 wh

```

1534212

```

int Opción;
int VOTOS [ ];
int VOTO;
{ if (voto < min) || (voto > max) } }
cout ~
cin >> ~
do (escrutinio) / /
white
int
salario?
cin >> salario;
PF (Salario) << resultado;
IFC salario - 100%
100% / /
10

```

```

void printList()
{
    cout << "Current list: ";
    for (int i = 0; i < num; i++)
        cout << arr[i] << " ";
    cout << endl;
}

int main()
{
    int n, m;
    cout << "Enter number of elements: ";
    cin >> n;
    cout << "Enter min and max values: ";
    cin >> m >> n;
    cout << "Enter array elements: ";
    for (int i = 0; i < n; i++)
        cin >> arr[i];
    cout << endl;
    cout << "List of elements: ";
    printList();
}

```

---

Implementation of Queue using Array

```

class Queue
{
public:
    int front, rear;
    int size;
    int *arr;
    Queue(int s)
    {
        size = s;
        arr = new int[s];
        front = -1;
        rear = -1;
    }
    void enqueue(int x)
    {
        if (isFull())
            cout << "Queue is full" << endl;
        else
        {
            arr[++rear] = x;
            if (front == -1)
                front = 0;
        }
    }
    int dequeue()
    {
        if (isEmpty())
            cout << "Queue is empty" << endl;
        else
        {
            int x = arr[front];
            arr[front] = -1;
            if (front == rear)
                front = rear = -1;
            else
                front++;
            return x;
        }
    }
    bool isEmpty()
    {
        return (front == -1);
    }
    bool isFull()
    {
        return (front + 1 == rear + 1);
    }
};

Queue::Queue(int s)
{
    size = s;
    arr = new int[s];
    front = -1;
    rear = -1;
}

int main()
{
    Queue q(5);
    q.enqueue(10);
    q.enqueue(20);
    q.enqueue(30);
    q.enqueue(40);
    cout << q.dequeue() << endl;
    cout << q.dequeue() << endl;
}

```

---

Implementation of Stack using Array

```

class Stack
{
public:
    int top;
    int size;
    int *arr;
    Stack(int s)
    {
        size = s;
        arr = new int[s];
        top = -1;
    }
    void push(int x)
    {
        if (isFull())
            cout << "Stack is full" << endl;
        else
        {
            arr[++top] = x;
        }
    }
    int pop()
    {
        if (isEmpty())
            cout << "Stack is empty" << endl;
        else
            return arr[top--];
    }
    bool isEmpty()
    {
        return (top == -1);
    }
    bool isFull()
    {
        return (top + 1 == size);
    }
};

Stack::Stack(int s)
{
    size = s;
    arr = new int[s];
    top = -1;
}

int main()
{
    Stack s(5);
    s.push(10);
    s.push(20);
    s.push(30);
    cout << s.pop() << endl;
    cout << s.pop() << endl;
    cout << s.pop() << endl;
}

```

# Exercicis d'informàtica (103806)

Curs 2020-2021

Examen Parcial (23 de Novembre de 2020)  
Nom estudiant: DONALD BROCKE HUGS

Grup: NIU: 8163427

Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `True` en el cas que siguin iguals i `False` en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

$$98 + 3 = 101$$

RESULTAT = 126

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bumbo. Els cartons tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bumbo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartro` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 11 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

• 5 nombres del 31 al 45.

El procediment rebrà un array (que serà el cartró) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- `int GenerarNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min-max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordene l'array a de dimensió dim de menor a major.

$$31 + 31 = 62$$

$$90 + 10 = \underline{100}$$

Exercici 3 (1 punt)

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingirà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`index`) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

NOTA: Recordeu que els números de les boles van de 1 a 75.

#### Exercici 4 (1 punt)

Fer una funció anomenada Escrutini per saber si un cartó ha aconseguit Bingo. La funció rebrà un array que serà el cartó, la dimensió del cartó i un array (boom) que contindrà informació dels nombres que han sortit. En l'array boom si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i False en cas contrari.

#### Exercici 6 (2,5 punts)

Fer una funció anomenada ImprimirComentari per imprimir un comentari sobre el

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenuPrincipal()
{
    cout << " --- Menu Principal --- " << endl;
    cout << "1.- Asignar cartones " << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Asignar cartons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
```

Inicialitzar el vector a de dimensió dim al valor v

```
int GenerarNombre(int min, int max, int a[], int dim)
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartons de dos jugadors anomenats cartoPlayer1 i cartoPlayer2. Declarar també un array, anomenat boom, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment MenuPrincipal () i llegir l'opció escollida.

4. Si l'opció és 1, generar dos cartons (un per a cada jugador) seguint els passos següents:

- 4.1. Inicialitzar els cartons dels jugadors tot a 0 amb el procediment InicialitzarArray.
- 4.2. Utilitzar el procediment CrearCarto per crear els cartons del jugador 1 i del jugador 2.

4.3. Comprovar si els dos cartons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.

5. Si l'opció és 2, implementa el joc seguint els passos següents:

- 5.1. S'haurà de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
- 5.2. Inicialitzar l'array boom a 0 amb el procediment InicialitzarArray.

5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.

5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els nombres de les boles que han sortit al llarg de la partida.

5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.

5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3

6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa..."
8. Qualsevol altra opció, escriure el missatge: "Opció no permetuda".
9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0.5 punts)**

Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `Nom_Jugador` (cadena de 50 caràcters).

Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeix l'identificador del cartro i el nom del jugador separant-los amb el caràcter ‘.

**Exercici 9 (1 punt)**

Escriure la funció `Elevar` que retorni el resultat d'elevat un nombre al quadrat. La funció tindrà 2 paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclòs en l'intervall [-46.340,46.340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada `Es_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

1528461

100 → 37

Vote 37

Resultat [37] ++;

low frequency

12535

in simple *Monotropa* (not root - especially *spadix*) growing in  
moist shade

( $\sigma = \sigma_{\text{max}}$ )  
 $\sigma_{\text{max}} = 0.0001$

( $\sigma = \sigma_{\text{max}}$ )  
 $\sigma_{\text{max}} = 0.0001$

( $\sigma = \sigma_{\text{max}}$ )

( $\sigma = \sigma_{\text{max}}$ )

( $\sigma = \sigma_{\text{max}}$ )

$\sigma = 0$

$\sigma = \sigma_{\text{max}}$

$\sigma = 0$

o winter  
flock

1000' - 10000' (1000')  
L. (Wade) to north side  
lowered - 10000'  
and pole  
and (Wade) to  
winter) the  
3000'

s few

10000'

10000' - 10000'

10000' - 10000'  
10000' - 10000'

10000' - 10000'

10000' - 10000'

10000' - 10000'  
10000' - 10000'

10000' - 10000'

10000'

10000' - 10000'

10000'

10000'

10000'

10000' - 10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

10000'

# Examen Parcial (23 de Novembre de 2020)

Curs 2020-2021  
Grup:  
NIU: 2026813

Nom estudiant: Hèrcules Gómez Pineda

**Exercici 2 (1 punt)**  
Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `True` en el cas que siguin iguals i `False` en cas contrari.

**NOTA:** Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$800 \div 20 = 40$$

Resultat : 2

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartells tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartell. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

**Exercici 1 (1 punt)**

Fer un procediment anomenat `CrearCartell` per a generar un cartell. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1'1 al 15.
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

• 5 nombres del 16 al 30.

• 5 nombres del 31 al 45.

El procediment rebrà un array (que serà el cartell) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

**UTILITZEU (no implementar) els següents procediments o funcions:**

- `int GenerarNombre(int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena(int a[], int dim)`  
Ordena l'array a de dimensió dim de menor a major.

$$7 \times 4 = 28$$
$$28 + 120 = \underline{148}$$

**Exercici 3 (1 punt)**

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contindrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`index`) és un 1, voldrà dir que la bola corresponent ha sortit; si el és un 0, voldrà dir que aquella bola no ha sortit.

**NOTA:** Recordeu que els números de les boles van de 1 a 75.

**Exercici 4 (1 punt)**  
Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contindrà informació dels noms que han sortit. En l'array bombó si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola corresponent ha sortit si el és un 0, voldrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

**Exercici 6 (2.5 punts)**  
Suposeu que tenu els següents procediments i funcions ja implementats:

```
void MenuPrincipal()
{
    cout << " --- Menu Principal --- " << endl;
    cout << "1.- Asignar Cartrons " << endl;
    cout << "2.- Jugar " << endl;
    cout << "3.- Marcador " << endl;
    cout << "4.- Sortir " << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Assignar cartrons, 2. Jugar, etc.

void InicialitzarArray(int a[], int dim, int v)

Inicialitza el vector a de dimensió dim al valor v

int GenerarNombre (int min, int max, int a[], int dim)

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim.

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

1. Declarar (on corresponguï) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on corresponguï) els arrays necessaris per representar els cartrons de dos jugadors anomenats cartroPlayer1 i cartroPlayer2. Declarar també un array, anomenat bombo, per controlar quins noms han sortit i quins no.
3. Utilitzar el procediment MenuPrincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartrons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CrearCartro per crear els cartrons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartrons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartrons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els noms de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
  6. Si l'opció és 3, imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
  7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa . . .".
  8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
  9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigua la 4.



**Exercici 7 (0.5 punts)**

Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `Nom_Jugador` (cadena de 50 caràcters). Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeix l'identificador del cartró i el nom del jugador separant-los amb el caràcter ‘‘.

**Exercici 9 (1 punt)**

Escriure la funció `Elevar` que retorni el resultat d'elevar un nombre al quadrat. La funció tindrà dos paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornara un 1 si s'ha pogut realizar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'intervall [-46.340,46.340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada `Es_MatPositiva` que rebi com a paràmetres: una matrú d'enters de 4 columnes, el número de files de la matrú, i el número de columnes de la matrú, i retorna 1 si tots els valors de la matrú són positius o zero, i un 0 en cas contrari.

1528864

1 in 5 \$  
2 ✓ 6 check  
3 ✓ check ✓  
4 9  
50

total = false

if G ! total

total = true

switch (dia)

case 1 :

; break;

typedef struct

{

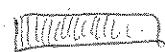
};

A B



1524824

votacion de



150

espiral:



12

$$x = \frac{b}{a} \quad a \neq 0$$



# onaments d'informàtica (TUSUB)

Curs 2020-2021

Examen Parcial (23 de Novembre de 2020)

Nom estudiant: Nancy Stallinga Geno

Grup: NIU: 441 8379

Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `true` en el cas que siguin iguals i `false` en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.) Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherència entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el tenu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartons tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els números del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadaescú.

**Exercici 1 (1 punt)**  
Fer un procediment anomenat `CrearCartro` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del l'1 al 15.
- 5 nombres del l'16 al 30.
- 5 nombres del 31 al 45.
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

**UTILITZEU** (no implementar) els següents procediments o funcions:

- `int GenerarNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min,max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordena l'array a de dimensió dim de menor a major.

$$5 + 14 = 23$$

$$23 \times 10 = \underline{120}$$

Resultat: 54  
$$94 + 14 = 108$$
$$108 \div 2 = 54$$

#### Exercici 4 (1 punt)

Fer una funció anomenada Escrutini per saber si un cartó ha aconseguit Bingo. La funció rebrà un array que serà el cartó, la dimensió del cartó i un array (bombo) que contindrà informació dels noms que han sortit. En l'array bombo si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola correspondent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

#### Exercici 6 (2,5 punts)

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void Menuprincipal()
{
    cout << " --- Menu Principal --- " << endl;
    cout << "1. - Asignar cartones " << endl;
    cout << "2. - Jugar " << endl;
    cout << "3. - Marcador" << endl;
    cout << "4. - Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Assignar cartons, 2. Jugar, etc.

voi void InicialitzarArray(int a[], int dim, int v)

Inicializa el vector a de dimensió dim al valor v

```
int GenerarNombre(int min, int max, int a[], int dim)
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartons de dos jugadors anomenats cartonPlayer1 i cartonPlayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment Menuprincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CrearCarto per crear els cartons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartons són diferents utilitzant la funció AreTheyEqual. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els noms de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
  6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
  7. Si l'opció es 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa..."
  8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
  9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0,5 punts)**

Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `nom_Jugador` (cadena de 50 caràcters).  
Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeix l'identificador del cartró i el nom del jugador separant-los amb el caràcter ‘-’.

**Exercici 9 (1 punt)**

Escriure la funció `Elevar` que retorna el resultat d'elevat un nombre al quadrat. La funció tindrà 2 paràmetres de tipus `int`. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realitzar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'intervall [-46,340,46,340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada `Es_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el numero de columnnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

(525836)

$$1000 \quad | \quad \begin{array}{r} 101 \\ 0'000 \end{array}$$

$$\frac{1}{101}$$

$$\frac{1}{100} \approx 0'001$$

exp min 10      1's

100+

$$\begin{array}{ll} 11'5 & 50t \rightarrow 15 \\ 11'5 & 78t \rightarrow 12'5 \\ 10t & \rightarrow 71 \end{array}$$

$$X = \frac{1}{a}$$

$$10\% \text{ do } 10 \Leftrightarrow \frac{10 \cdot 10}{100} = 1$$

$$\frac{10 \cdot 25}{100} = 2'5$$



Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `True` en el cas que siguin iguals i `False` en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$225 + 25 = 250$$

$$250 - 125 = 125$$

Resultat: 125

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherència entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartells tenen un total de 25 números. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els números del nostre cartell. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

#### Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartell` per a generar un cartell. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 11 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 31 al 45,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartell) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

`UTILIZEU` (no implementar) els següents procediments o funcions:

- `int GeneraNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordena l'array a de dimensió dim de menor a major.

$$14 + 7 = 18$$

$$18 \times 4 = \underline{\underline{72}}$$

#### Exercici 3 (1 punt)

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`index`) és un 1, vindrà dir que la bola corresponent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

NOTA: Recordeu que els números de les boles van de 1 a 75.

Fer una funció anomenada Escrutini per saber si un cartó ha aconseguit Bingo. La funció rebrà un array que serà el cartó, la dimensió del cartó i un array (bombo) que contindrà informació dels nombres que han sortit. En l'array bombo si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola correspondent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

Exercici 6 (2,5 punts)  
Suposeu que teniu els següents procediments i funcions ja implementats:

```

void MenuPrincipal()
{
    cout << "-->> Menú Principal ---" << endl;
    cout << "1.- Asignar cartones" << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}

```

Mostra per pantalla un menú amb les opcions del codi: 1. Asignar cartones, 2. Jugar, etc.

```

void InicialitzarArray(int a[], int dim, int v)
{
    int GenerarNombre(int min, int max, int a[], int dim);
    Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim
}

```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:  
1. Declarar (on corresponguï) les constants necessàries per tal que el programa sigui fàcilment modificable.

2. Declarar (on corresponguï) els arrays necessaris per representar els cartons de dos jugadors anomenats cartoPlayer1 i cartoPlayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.

3. Utilitzar el procediment MenuPrincipal () i llegir l'opció escollida.

4. Si l'opció és 1, generar dos cartons (un per a cada jugador) seguint els passos següents:

4.1. Inicialitzar els cartons dels jugadors tot a 0 amb el procediment InicialitzarArray.

4.2. Utilitzar el procediment CrearCartro per crear els cartons del jugador 1 i del jugador 2.

4.3. Comprovar si els dos cartons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.

5. Si l'opció és 2, implementa el joc seguint els passos següents:

5.1. S'haurà de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.

5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.

5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.

5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els nombres de les boles que han sortit al llarg de la partida.

5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.

5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3

6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.

7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa...."

8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".

9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



Declarar un nou tipus de dades, TCartró, com un registre amb els camps: Ident\_Cartró (valor enter), Numercs (array de 25 enters) i Nom\_Jugador (cadena de 50 caràcters).

Fer un procediment anomenat Imprimir que rebi com a paràmetre un registre del tipus TCartró i imprimeix l'identificador del cartró i el nom del jugador separant-los amb el caràcter ','.

**Exercici 9 {1 punt}**  
Escriviu la funció El·levar que retorni el resultat d'elevat un nombre al quadrat. La funció tindrà paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realitzar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

**NOTA:** El valor de la base ha d'estar inclos en l'interval [-46.340,46.340] per evitar overflow.

### Exercici 8 (1 punt)

Fer una funció anomenada Es\_MatPositiva que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorni 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `True` en el cas que siguin iguals i `False` en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$56 \times 1 = 56$$
$$20 + 56 = 76$$

**Important:** Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bumbo. Els cartons tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bumbo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caindrà portar el compte de quantes partides ha guanyat cadascú.

#### Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartro` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 31 al 45,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

**UTILITZEU** (no implementar) els següents procediments o funcions:

- `int GeneraNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordene l'array a de dimensió dim de menor a major.

$$31 + 48 = 79$$
$$79 + 86 = \underline{165}$$

#### Exercici 3 (1 punt)

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contindrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`index`) és un 1, voldrà dir que la bola corresponent ha sortit i si el ès un 0, voldrà dir que aquella bola no ha sortit.

NOTA: Recordeu que els números de les boles van de 1 a 75.

**Exercici 4 (1 punt)**  
 Fer una funció anomenada `Escrutini` per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (`bombo`) que contingrà informació dels nombres que han sortit. En l'array `bombo` si el valor d'una determinada posició (`index`) és un 1, vindrà dir que la bola correspondent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

La funció retornarà `True` en el cas que s'hagi aconseguit Bingo i `False` en cas contrari.

**Exercici 6 (2,5 punts)**  
 Suposeu que teniu els següents procediments i funcions ja implementats:

```
void InicialitzarArray(int a[], int dim, int v)
{
    cout << "---- Menu Principal ----" << endl;
    cout << "1.- Asignar cartrons" << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Asignar cartrons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
```

Inicialitza el vector `a` de dimensió `dim` al valor `v`

```
int GenerarNombre(int min, int max, int a[], int dim)
```

Retorna un enter aleatori entre `min` i `max`, controlant que no estigui a l'array `a` de dimensió `dim`

Fer un programa complet (declaracions globals i funció `main()`) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartrons de dos jugadors anomenats `cartróJugador1` i `cartróJugador2`. Declarar també un array, anomenat `bombo`, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment `MenuPrincipal()` i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartrons dels jugadors tot a 0 amb el procediment `InicialitzarArray`.
  - 4.2. Utilitzar el procediment `CreateCartró` per crear els cartrons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartrons són diferents utilitzant la funció `AreTheyEqual`. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartrons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array `bombo` a 0 amb el procediment `InicialitzarArray`.
  - 5.3. Generar un nombre per simular la bola amb la funció `GenerarNombre`.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció `ImprimirBolesSortides` per imprimir els nombres de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció `Escrutini` per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
  6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment `ImprimirComentari`.
  7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa. . .".
  8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
  9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0,5 punts)**  
Declarar un nou tipus de dades, TCartro, com un registre amb els camps: Ident\_Carro (valor enter), Numeros (array de 25 enters) i nom\_Jugador (cadena de 50 caràcters).

Fer un procediment anomenat Imprimir que rebi com a paràmetre un registre del tipus TCarro i imprimeix l'identificador del carro i el nom del jugador separant-los amb el caràcter ‘\_’.

**Exercici 9 (1 punt)**  
Escriure la funció Elevar que retorni el resultat d'elevar un nombre al quadrat. La funció tindrà paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si s'ha pogut realizar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar incòs en l'intervall [-46.340,46.340] per evitar overflow.

### Exercici 8 (1 punt)

Fer una funció anomenada Es\_MatPositiva que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

iostream

define m

def 150

Int Voter (int min, int max)

cout << "enter voter entre \" " << min << " " << max "

a >> a >> b;

while (a >> b)

4. PostStream

int main()

switch (MemOp)

case 1:

Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen els mateus valors internos. La funció rebrà dos arrays i la seva dimensió. Retornarà `True` en el cas que siguin iguals i `False` en cas contrari.

**NOTA:** Podeu suposar que els dos arrays tenen la mateixa dimensió.

**Important:** Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherència entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el tenui a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo.

Els cartons tenen un total de 25 números. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els números del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

#### Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartro` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1 al 15, • 5 nombres del 46 al 60 i
- 5 nombres del 16 al 30, • 5 nombres del 61 al 75.
- 5 nombres del 31 al 45,

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

**UTILITZEU** (no implementar) els següents procediments o funcions:

- `int GenerarNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordena l'array a de dimensió dim de menor a major.

$$10 \times 10 = \underline{100}$$

$$100 + 300 = \underline{400}$$

**Resultat :** 999

$$999 - 333 = 666$$

$$666 + 333 = 999$$

#### Exercici 3 (1 punt)

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingdrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`index`) és un 1, vindrà dir que la bola corresponent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

**NOTA:** Recordeu que els números de les boles van de 1 a 75.

Fer una funció anomenada Escrutini per saber si un cartó ha aconseguit Bingo. La funció rebrà un array que serà el cartó, la dimensió del cartó i un array (bombo) que contindrà informació dels nombres que han sortit. En l'array bombó si el valor d'una determinada posició (*index*) és un 1, vindrà dir que la bola corresponent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

La funció retornarà `True` en el cas que s'hagi aconseguit Bingo i `False` en cas contrari.

**Exercici 6 (2.5 punts)**  
Suposeu que teniu els següents procediments i funcions ja implementats:

```
void InicialitzarArray(int a[], int dim, int v)
{
    cout << "---- Menu Principal ---" << endl;
    cout << "1.- Asignar cartons" << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Asignar cartons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
{
    cout << "---- Menu Principal ---" << endl;
    cout << "1.- Asignar cartons" << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartons de dos jugadors anomenats cartoPlayer1 i cartoPlayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment `MenuPrincipal()` i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartons dels jugadors tot a 0 amb el procediment `InicialitzarArray`.
  - 4.2. Utilitzar el procediment `CrearCarto` per crear els cartons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartons són diferents utilitzant la funció `ArraysIguals`. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment `InicialitzarArray`.
  - 5.3. Generar un nombre per simular la bola amb la funció `GenerarNombre`.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció `ImprimirBolesSortides` per imprimir els nombres de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció `Escrutini` per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
  6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment `ImprimirComentari`.
  7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa..."
  8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
  9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



Declarar un nou tipus de dades, TCartró, com un registre amb els camps: Ident\_Cartró (valor enter), Numeros (array de 25 enters) i Nom\_Jugador (cadena de 50 caràcters).

Fer un procediment anomenat Imprimir que rebi com a paràmetre un registre del tipus TCartró i imprimeix l'identificador del cartró i el nom del jugador separant-los amb el caràcter ‘\_’.

Exercici 9 (1 punt)

Escriure la funció Elevar que retorni el resultat d'elevat un nombre al quadrat. La funció tindrà paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'intervall [-46.340,46.340] per evitar overflow.

#### Exercici 8 (1 punt)

Fer una funció anomenada Es\_MatPositiva que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

10 int ComprobacionPos9 (int anys, int mitjana)  
 int punts;  
 if (exp < cipmin.(2.5/100) ) { // keep expression (10/100)  
 punts = 1;  
 } else { if (exp > 50/100) {  
 } //

8 int Recompte (int rotacions, int numbers,  
 rotatedes [numbers]  
 escriptiu [numAgents])

$$\frac{10 \cdot 1}{100} = 12$$

10.1  
 100  
 12

Recompte (veleció, numbers, escriptiu)

(puntuació, numbers, 0)

(veleció, numbers, 0)

~~100~~

veleció = voter (numbers, max es.)

(+ + numbers, 0 = ?) set

veleció

: 0 = 8 + 0.0 { + veler (numbers, max es.) (0 = q + x, 0) f.

0 + 0/8 = + 8 0 (0)/(q+) = x

(x 8 + x) / q + x / 8 + x = numbers, 0

③ void inicializarMatriz (int &E[], int dim) {  
④     int i;  
    i = 0;  
    while (i < dim) { E[i] = 0; }  
    if (i == dim) { return 0; }  
    else { return 1; }  
}

void Recompete (int &Votaciones[], int dimVotaciones);

~~Votación~~

⑧ typedef struct {

    char Nom [50];  
    char Empresa [30];  
    int Vot;

} Tclient;

int main ()

⑨ int EquacioPrimerGrado (int a, int b, int &x)

    if (a \* x + b == 0) { return 1; }  
    x = (-b) / (a);

    else { if (

$$ax + b = 0$$

) { cout << "Solución: " << x << endl; }

else

    return 0;

    return 1;

(now for 'num pur') NOTA: PUN

⑩

5

Recompte (int. Votacion) lot 3.

300

卷之三

卷之三

1

4

2

1

1

```
int IndexMaximArray (const int codis[], int dim)
```

100

for  $i = 0$  to  $n - 1$

1

value = codisitJ;

H ( value 1 > value 2 )

250

$$M_{\mathrm{box}} = \tilde{c}^{-1} \tilde{t}$$

1

104 man ( )

Int. Vistasons [INCUBENTS]; (at open); (at zero);  
Int. Resonate [INCORPORATED]; (not sent); (at i)

IncubatorArray ( lessons, clients, 0 );  
MainPrincipal();

CIO 25 8PC 08

(ode)  $\dot{y} = f(t, y)$

卷之三

Old > Ident

WILHELM ROBERT ULLMANN (IDENT=HOLZLEINER) /

Abstacos [Ident] = Votar (101, 112)

2222253

3 void InicializarArray (int codis[], int Dimm, int valor)

```
    {
        int i;
        do
            {cout << "Escribe un valor entre " << min << " y ";
            << max << endl;
            cin >> valor;
        } while ((valor >= min) && (valor <= max));
        codis[i] = valor;
    }
```

2

int Mayor (int min, int max)

```
    {
        int valor;
        do
            {cout << "Escribe un valor entre " << min << " y ";
            << max << endl;
            cin >> valor;
        } while ((valor < min) || (valor > max));
        cout << "El valor es correctamente introducido";
    }
```

4.

```
int cuentaZero (int codis[], int Dimm)
{
    int i;
    int zero = 0;
    for (i = 0; i < Dimm; i++)
    {
        if (codis[i] == 0)
            zero = 1;
    }
    return zero;
}
```

La funció rebrà dos arrays i la seva dimensió. Retornarà True en el cas que siguin iguals i False en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.) Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherència entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartells tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartell. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

**Exercici 1 (1 punt)**

Fer un procediment anomenat CrearCartell per a generar un cartell. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 11 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 31 al 45,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartell) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- int GenerarNombre (int min, int max, int a[], int dim)
- void Ordena (int a[], int dim)

Ordene l'array a de dimensió dim de menor a major.

$$239 - 100 = 139$$
$$239 + 100 = \boxed{339}$$

$$326 \times 2 = 625$$
$$625 + 152 = 777$$

**Exercici 3 (1 punt)**

Fer un procediment anomenat ImprimirBolesSortides per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingirà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (*index*) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

NOTA. Recordeu que els números de les boles van de 1 a 75.





Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident` – Cadira  
(valor enter), `Numeros` (array de 25 enters) i `Nom_Jugador` (cadena de 50 caràcters).

Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeixi l'identificador del cartó i el nom del jugador separat-los amb el caràcter '-'.

Escriviu la funció `Elevar` que retorna el resultat d'elevar un nombre al quadrat. La funció té dos paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realitzar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'intervall [-46.340, 46.340] per evitar overflow.

### Exercici 8 (1 punt)

Fer una funció anomenada `Es_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

1527046

$$ax+b=0$$

$$x = \frac{-b}{a} \quad \text{if } a \neq 0 \quad \text{NS}$$

$$\text{if } a = 0$$

extra value of the



## Examen Parcial (23 de Noviembre de 2020)

Nombre estudiante: Alejo Serrano Alarcón

Grupo: NIU: 2A 38560

Importante: Recuerda que hay que dar las mejores soluciones posibles en cada ejercicio. Además de funcionar correctamente, el código debe estar bien programado (código claro, con las instrucciones más adecuadas, sin operaciones ni variables innecesarias, etc.)

Las funciones y procedimientos de las preguntas de la 1 a la 6 forman parte de un único programa y, por tanto, debe haber coherencia entre sus definiciones. Y la forma en que se utilizan en otras preguntas. El contexto del programa a desarrollar, lo tenéis a continuación:

El Bingo de 75 bolas es un juego de azar donde hay 75 bolas numeradas del 1 al 75 dentro del bombo. Los cartones tienen un total de 25 números. En este tipo de bingo, se van sacando bolas del bombo y se gana un premio cuando han salido todos los números de nuestro cartón. Esto se conoce como bingo o full House.

Queremos hacer un programa que permita jugar al Bingo a dos jugadores. Los jugadores podrán jugar tantas partidas como quieran y habrá que llevar la cuenta de cuántas partidas ha ganado cada uno.

### Ejercicio 1 (1 punto)

Hacer un procedimiento llamado `CrearCartón` para generar un cartón. Se deben generar 25 números del 1 al 75. Se debe asegurar que haya:

- 5 números del 1 al 15,
- 5 números del 16 al 30,
- 5 números del 31 al 45,
- 5 números del 46 al 60 y
- 5 números del 61 al 75.

el procedimiento recibirá un array (que será el cartón) y la dimensión del array. Devolverá el array lleno con los números ordenados de menor a mayor.

UTILIZA (no implementar) los siguientes procedimientos o funciones:

- `int GenerarNombre (int min, int max, int a [], int dim)`  
Genera un número aleatorio entre min y max, y revisa que no esté en el array a de dimensión dim.
- `void Ordenar (int a [], int dim)`  
Ordenar el array a de dimensión dim de menor a mayor.

$$50 \times 7 = 350$$

$$350 - 63 = \boxed{287}$$

### Ejercicio 2 (1 punto)

Hacer la función llamada `ArraysIguales` para saber si dos arrays tienen todos los valores idénticos. La función recibirá dos arrays y su dimensión, devolverá `True` en caso de que sean iguales y `False` en caso contrario.

NOTA: Podéis suponer que los dos arrays tienen la misma dimensión.

$$40 + 68 = 108$$

Resultado = 108

### Ejercicio 3 (1 punto)

Hacer un procedimiento llamado `ImprimirBolasSorteadas` para imprimir los números de las bolas que ya han salido. El procedimiento recibirá un array y su dimensión. El array contendrá información sobre si un número ha salido o no. Si el valor de una determinada posición (índice) es un 1, querrá decir que la bola correspondiente ha salido y si el es un 0 querrá decir que aquella bola no ha salido.

NOTA: Recuerda que los números de las bolas van de 1 a 75.

#### Ejercicio 4 (1 punto)

Hacer una función llamada Escrutinio para saber si un cartón ha conseguido Bingo. La función recibirá un array que será el cartón, la dimensión del cartón y un array (bombo) que contendrá información de los números que han salido. En el array bombó si el valor de una determinada posición (índice) es un 1, querrá decir que la bola correspondiente ha salido y si el es un 0, querrá decir que aquella bola no ha salido.

La función devolverá `True` en caso de que se haya conseguido Bingo y `False` en caso contrario.

#### Ejercicio 6 (2.5 puntos)

Supongamos que tiene los siguientes procedimientos y funciones ya implementados:

```
void MenúPrincipal ()  
{ cout << "---- Menú Principal ----" << endl;  
cout << "1.- Asignar cartones" << endl;  
cout << "2.- Jugar" << endl;  
cout << "3.- Marcador" << endl;  
cout << "4.- Salir" << endl; }
```

Muestra por pantalla un menú con las opciones del código: 1. Asignar cartones, 2. Jugar, etc.

```
int GenerarNombre (int min, int max, int a [], int dim)
```

```
void InicializarArray (int a [], int dim, int v)
```

Inicializa el vector a de dimensión dim al valor v

Hacer un entero aleatorio entre min y max, controlando que no esté en el array a de dimensión dim

Hacer un programa completo (declaraciones globales y función main ()) que siga los siguientes pasos:

- Declarar (donde corresponda) las constantes necesarias para que el programa sea fácilmente modificable.
- Declarar (donde corresponda) los arrays necesarios para representar los cartones de dos jugadores llamados cartónPlayer1 y cartónPlayer2. Declarar también un array, llamado bombo, para controlar qué números han salido y cuáles no.
- Utilizar el procedimiento MenúPrincipal () y leer la opción elegida.
- Si la opción es 1, generar dos cartones (uno para cada jugador) siguiendo los pasos siguientes:
  - Inicializar los cartones de los jugadores todo a 0 con el procedimiento InicializarArray.
- Utilizar el procedimiento CrearCartón para crear los cartones del jugador 1 y del jugador 2.
- Comprobar si los dos cartones son diferentes utilizando la función ArraysIguals. En caso de que sean iguales volver a generar uno de los dos.
- Si la opción es 2, implementa el juego siguiendo los siguientes pasos:
  - Se deberá comprobar que se han generado los cartones (ha entrado en la opción 1). En caso contrario, dar un mensaje de error y volver al menú principal sin hacer nada más.
- Inicializar el array bombo a 0 con el procedimiento InicializarArray.
- Generar un número para simular la bola con la función GenerarNombre.
- Imprimir el número que ha salido y utilizar la función ImprimirBolasSortidas para imprimir los números de las bolas que han salido a lo largo de la partida.
- Utilizar la función Escrutinio para comprobar si alguno de los dos jugadores ha conseguido hacer Bingo.
- En caso de que no haya ningún ganador volver al punto 5.3
- Si la opción es 3, imprimir el marcador de las partidas que se han jugado y mostrar el mensaje generado por el procedimiento ImprimirComentari.
- Si la opción es 4, imprimir el marcador final y mostrar el mensaje: "Saliendo del programa..."
- Cualquier otra opción, escribir el mensaje: "Opción no permitida".
- Repetir los pasos 3 a 9, hasta que la opción del menú escogida sea la 4.



**Ejercicio 7 (0.5 puntos)**

Declarar un nuevo tipo de datos, `TCarton`, como un registro con los campos: `Ident_Carton` (valor entero), `Numeros` (array de 25 enteros) y `Nombre_Jugador` (cadena de 50 caracteres).  
Hacer un procedimiento llamado `Imprimir` que reciba como parámetro un registro del tipo `TCarton` e imprima el identificador del cartón y el nombre del jugador separandolos con el carácter `,`.

**Ejercicio 9 (1 punto)**

Escribir la función `Elevar` que devuelva el resultado de elevar un número al cuadrado. La función tendrá 2 parámetros de tipo int. El primero será la base y el segundo será el resultado de la operación en caso de que se pueda realizar. La función devolverá un 1 si se ha podido realizar la operación y un 0 en caso contrario.

NOTA:El valor de la base debe estar incluido en el intervalo [-46.340,46.340] para evitar overflow.

**Ejercicio 8 (1 punto)**

Hacer una función llamada `Es_MatPositiva` que reciba como parámetros: una matriz de enteros de 4 columnas, el número de filas de la matriz, y el número de columnas de la matriz, y devuelva 1 si todos los valores de la matriz son positivos o cero, y un 0 en caso contrario.

S29287

1) 12 gents,  $\{1\ldots 12\}$   
 150 clients,  $\{0\ldots 149\}$

```

typedef struct {
    - int votaciones[12];
} Votaciones;

typedef struct {
    - int escrutini[150];
} Escrutini;
    
```

2) int Vota(int min, int max)

```

{
    int x;
    cout << "Intra valor entre " << min << " i " << max;
    cin >> x;
    while (x < min || x > max) (x < min || x > max) {
        cout << "Error: valor fora de l'intervall";
        cin >> x;
    }
    cout << "Vot enre's correcte";
    return x;
}
    
```

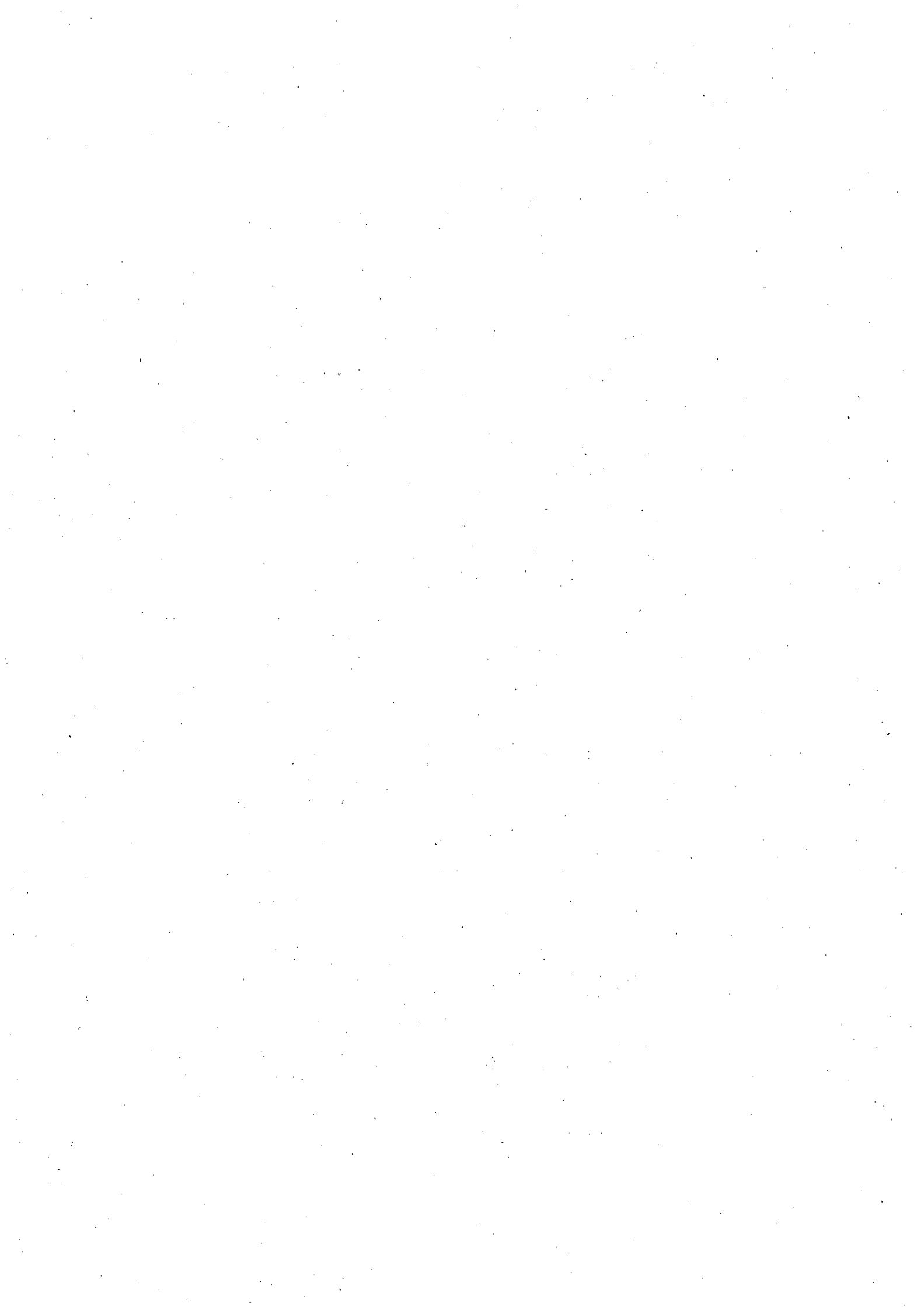
(full) esborrany



1529287

```
if (z == 0) {  
    g->IndexMaximArray(resultat.resultat, Agents(totals))  
    cout << "El comercial guanyador es el " << g->  
    if (z == 1) {  
        cout << "Encara no s'ha fet el recompte de vots";  
  
default:  
case 4:  
    return 0;  
    break;  
default:  
    cout << "Opció no reconeguda";  
    cin >> w;
```

full estabancny



contar algoritmos i nota;  
cifrados > nota; int points=0;

1529328

if (ans>=15)  
points = points + 5;

else if (ans>=12's)  
points = points + 3

if (nota > 9)  
points = points + 5;  
else  
if (nota > 7) & (nota <= 9)  
points = points + 3;

40

50 - 5400%

33%

$$\frac{150}{200} = 75\% \text{ of}$$



**Examen Parcial (23 de Novembre de 2020)**

Grup:  
NIU: 9663721

Nom estudiant: Aixequero Ripera

Per la funció anomenada `ARRAYS_IGUALS` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `True` en el cas que siguin iguals i `false` en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

**Important:** Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartrons tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Voleu fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar quantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

**Exercici 1 (1 punt)**

Fer un procediment anomenat `CrearCarto` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1'1 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- `int GenerarNombre(int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordena l'array a de dimensió dim de menor a major.

$$70 + 70 = 140$$
$$140 - 3 = \underline{137}$$

**Exercici 3 (1 punt)**

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contindrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`index`) és un 1, voldrà dir que la bola corresponent ha sortit; si el és un 0, voldrà dir que aquella bola no ha sortit.

NOTA. Recordeu que els números de les boles van de 1 a 75.

$$80 + 120 = 200$$
$$200 \div 10 = 20$$

**Resultat: 20**

Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contingrà informació dels nombres que han sortit. En l'array bombó si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola correspondent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

Suposeu que teniu els següents procediments i funcions ja implementats:

```

void Menuprincipal()
{
    cout << " --- Menu Principal --- " << endl;
    cout << "1.- Asignar cartones " << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}

```

Mostra per pantalla un menú amb les opcions del codi: 1. Assignar cartons, 2. Jugar, etc.

```

void InicialitzarArray(int a[], int dim, int v)
{
    int GenerarNombre(int min, int max, int a[], int dim)
    {
        // Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim
        // Inicialitza el vector a de dimensió dim al valor v
    }
}

```

int GenerarNombre(int min, int max, int a[], int dim)

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Inicialitza el vector a de dimensió dim al valor v

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

1. Declarar (on corresponguï) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on corresponguï) els arrays necessaris per representar els cartrons de dos jugadors anomenats cartroplayer1 i cartroplayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment Menuprincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartrons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CrearCartró per crear els cartrons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartrons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartrons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els nombres de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
  6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
  7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa..."
  8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
  9. Repetir els passos 3 a 9, fins que l'opció del menú es volga a seguir la 4.



Declarar un nou tipus de dades: `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `nom_Jugador` (cadena de 50 caràcters).

Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeixi l'identificador del cartó i el nom del jugador separant-lo amb el caràcter ‘,’.

Escriviu la funció `Elevar` que retorna el resultat de elevar un nombre al quadrat. La funció tindrà dos paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si s'ha pogut realizar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'interval [-46,340; 46,340] per evitar overflow.

### Exercici 8 (1 punt)

Fer una funció anomenada `Es_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

## Ejercicio 1

1424857

12 agentes comerciales  $\rightarrow$  101 - 112 (COMERCIAL)

150 clients q' votarán

## Ejercicio 6

int IndexMaxim Array ( int enteros [ ], int dime )

{

int i;

int index;

int maxim;

maxim = enteros [0];

index = 0;

for ( i=0; i < dime; i++ )

{ if ( enteros[i] > maxim )

{

enteros [ ] = i;

index = i;

}

} return index;

}

$$5/9 = x \quad //$$

1st queue ( food to food )

```

void Recompute ( int V[3], int dimV, int E[3] )
{
    int i;
    for ( i=0 ; i<dimV ; i++ )
    {
        E[i % IDCOMMERCIAL] += V[i];
    }
}

int ComplicatedOps ( float avgExp, float minExp,
                     float maxExp, float maxAssigns )
{
    if ( avgExp < 0 )
        return -1;
    else if ( avgExp > 2 * maxAssigns )
        return 1;
    else if ( avgExp - minExp < 0 )
        return 0;
    else
        return 2;
}

```

(b)  $\text{int } \text{EquationPrimeGrou}( \text{float } a, \text{float } b )$

$\phi = \frac{1}{\phi}$

use  $a = 0$

use  $b = 0$

use  $a = 1$

use  $b = 1$

use  $a = 2$

use  $b = 2$

use  $a = 3$

use  $b = 3$

use  $a = 4$

use  $b = 4$

use  $a = 5$

use  $b = 5$

use  $a = 6$

use  $b = 6$

use  $a = 7$

use  $b = 7$

use  $a = 8$

use  $b = 8$

use  $a = 9$

use  $b = 9$

use  $a = 10$

use  $b = 10$

use  $a = 11$

use  $b = 11$

use  $a = 12$

use  $b = 12$

use  $a = 13$

use  $b = 13$

use  $a = 14$

use  $b = 14$

use  $a = 15$

use  $b = 15$

use  $a = 16$

use  $b = 16$

use  $a = 17$

use  $b = 17$

use  $a = 18$

use  $b = 18$

use  $a = 19$

use  $b = 19$

use  $a = 20$

use  $b = 20$

use  $a = 21$

use  $b = 21$

use  $a = 22$

use  $b = 22$

use  $a = 23$

use  $b = 23$

use  $a = 24$

use  $b = 24$

use  $a = 25$

use  $b = 25$

use  $a = 26$

use  $b = 26$

use  $a = 27$

use  $b = 27$

use  $a = 28$

use  $b = 28$

use  $a = 29$

use  $b = 29$

use  $a = 30$

use  $b = 30$

use  $a = 31$

use  $b = 31$

use  $a = 32$

use  $b = 32$

use  $a = 33$

use  $b = 33$

use  $a = 34$

use  $b = 34$

use  $a = 35$

use  $b = 35$

use  $a = 36$

use  $b = 36$

use  $a = 37$

use  $b = 37$

use  $a = 38$

use  $b = 38$

use  $a = 39$

use  $b = 39$

use  $a = 40$

use  $b = 40$

use  $a = 41$

use  $b = 41$

use  $a = 42$

use  $b = 42$

use  $a = 43$

use  $b = 43$

use  $a = 44$

use  $b = 44$

use  $a = 45$

use  $b = 45$

use  $a = 46$

use  $b = 46$

use  $a = 47$

use  $b = 47$

use  $a = 48$

use  $b = 48$

use  $a = 49$

use  $b = 49$

use  $a = 50$

use  $b = 50$

use  $a = 51$

use  $b = 51$

use  $a = 52$

use  $b = 52$

use  $a = 53$

use  $b = 53$

use  $a = 54$

use  $b = 54$

use  $a = 55$

use  $b = 55$

use  $a = 56$

use  $b = 56$

use  $a = 57$

use  $b = 57$

use  $a = 58$

use  $b = 58$

use  $a = 59$

use  $b = 59$

use  $a = 60$

use  $b = 60$

use  $a = 61$

use  $b = 61$

use  $a = 62$

use  $b = 62$

use  $a = 63$

use  $b = 63$

use  $a = 64$

use  $b = 64$

use  $a = 65$

use  $b = 65$

use  $a = 66$

use  $b = 66$

use  $a = 67$

use  $b = 67$

use  $a = 68$

use  $b = 68$

use  $a = 69$

use  $b = 69$

use  $a = 70$

use  $b = 70$

use  $a = 71$

use  $b = 71$

use  $a = 72$

use  $b = 72$

use  $a = 73$

use  $b = 73$

use  $a = 74$

use  $b = 74$

use  $a = 75$

use  $b = 75$

use  $a = 76$

use  $b = 76$

use  $a = 77$

use  $b = 77$

use  $a = 78$

use  $b = 78$

use  $a = 79$

use  $b = 79$

use  $a = 80$

use  $b = 80$

use  $a = 81$

use  $b = 81$

use  $a = 82$

use  $b = 82$

use  $a = 83$

use  $b = 83$

use  $a = 84$

use  $b = 84$

use  $a = 85$

use  $b = 85$

use  $a = 86$

use  $b = 86$

use  $a = 87$

use  $b = 87$

use  $a = 88$

use  $b = 88$

use  $a = 89$

use  $b = 89$

use  $a = 90$

use  $b = 90$

use  $a = 91$

use  $b = 91$

use  $a = 92$

use  $b = 92$

use  $a = 93$

use  $b = 93$

use  $a = 94$

use  $b = 94$

use  $a = 95$

use  $b = 95$

use  $a = 96$

use  $b = 96$

use  $a = 97$

use  $b = 97$

use  $a = 98$

use  $b = 98$

use  $a = 99$

use  $b = 99$

use  $a = 100$

use  $b = 100$

Recorre  
{ int i; aux;  
for (  
{  
    Votaciones [i] = aux;  
    Escrutini [aux-1] = Escrutini [aux-1] + 1;  
}

1446505



Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `True` en el cas que siguin iguals i `False` en cas contrari.

NOTA: Podieu suposar que els dos arrays tenen la mateixa dimensió.

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartrons tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

#### Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartro` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 11 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (*no implementa*) els següents procediments o funcions:

- `int GenerarNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordena l'array a de dimensió dim de menor a major.

$$90 - 35 = 55$$

$$55 \div 5 = \boxed{11}$$

#### Exercici 3 (1 punt)

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`index`) és un 1, valdrà dir que la bola corresponent ha sortit; si el és un 0, valdrà dir que aquella bola no ha sortit.

NOTA. Recordeu que els números de les boles van de 1 a 75.

$$\begin{aligned} 63 - 12 &= 51 \\ 51 \times 20 &= 1020 \\ \text{Resultat: } 1020 \end{aligned}$$

#### Exercici 4 (1 punt)

Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contindrà informació dels nombres que han sortit. En l'array bombó si el valor d'una determinada posició (*index*) és un 1, vindrà dir que la bola corresponent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

#### Exercici 6 (2,5 punts)

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenúPrincipal()
{
    cout << " --- Menú Principal --- " << endl;
    cout << "1.- Asignar cartrons " << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Mostra per pantalla un menú amb les options del codi: 1. Assignar cartrons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
{
    int GeneraNombre(int min, int max, int a[], int dim)
    {
        Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim
    }
}
```

Inicialitza el vector a de dimensió dim al valor v

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartrons de dos jugadors anomenats cartróPlayer1 i cartróPlayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment MenúPrincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartrons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CrearCartró per crear els cartrons del jugador 1 i del jugador 2.
- 4.3. Comprovar si els dos cartrons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartrons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els nombres de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
  6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
  7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa....".
  8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
  9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



Exercici / (0,5 punts)

Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `Nom_Jugador` (cadena de 50 caràcters).

Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeixi l'identificador del cartró i el nom del jugador separant-lo amb el caràcter ‘,’.

Exercici y (1 punt)

Escriure la funció `Elevar` que retorni el resultat de elevar un nombre al quadrat. La funció tindrà dos paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'intervall [-46.340,46.340] per evitar overflow.

Exercici 8 (1 punt)

Fer una funció anomenada `E.S_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorni 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

1526616

$$\begin{cases} a=0 \\ \text{Si } a \neq 0 \end{cases}$$

$$\frac{\partial}{\partial b} \neq 0$$

$$X = -\frac{b}{a} = 0$$

•

$$\begin{cases} a=1 \\ b>0 \end{cases}$$

$$\begin{cases} X+0=0 \\ X=0 \end{cases}$$

3) 0,5 lts.  
Unidimensional  
rep

1. May existas
2. dimension
3. valor unico

L'altitud de la montaña pasada con el 3º parámetro

4) Hiperboloide  
1 pt.  
1. May existas

1. Dimension
2. Punto si hay o  
punto si no hay o

5) 6 pt.  
1. Sobre Máximo y  
Mínimo

1. May existas
2. Dimension

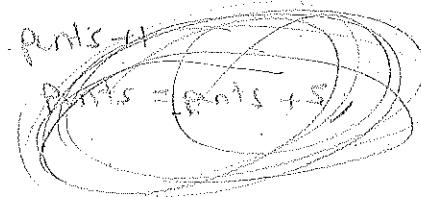
Resumen línea de p. máxim



# Index Minim Array (int v[], int dim)

1497995

```
int i;
int volmax = V[0];
int posmax = 0;
for (i=0; i<dim; i++)
    { if (V[i] > volmax)
        { volmax = V[i];
          posmax = i;
        }
    }
return posmax;
```



if ( $\alpha == 0$ )

$$\text{mín} \alpha = 10 \text{ años}$$

```
int CompruebaOpos (float anys, float mitjana)
```

```
{ int puntos;?
    int exp;
    if (exp > 15)
        return puntos;
    }
```

$$\checkmark 10 \text{ años}$$



$$10 \text{ años} +$$

~~if (exp > 12.5) && (exp <= 15)~~

$$\text{any} \alpha = 10$$

$$\text{any} \alpha 1 = \text{any} \alpha + \text{o's. om}$$

~~if (exp > 10) && (exp <= 12.5)~~

~~if (mitjana > 9)~~

~~if (mitjana >= 7) && (mitjana <= 9)~~

whale (ape)

defeat it? Could we open up some new pathways?"

CASE 4 : baseball

“Police” complain of gunplay as as 1811.

ISLAMABAD, Pakistan (CNN) -- The United States has agreed to allow the U.S. military to conduct surveillance flights over Pakistan, a senior Pakistani official said Saturday.

Case 3: If ( $w_1 = \text{NAMELESS}$ )  
"Dancer"  $\Rightarrow$  "Enebra"  $\Rightarrow$  so not for all  $(\neg \forall x \text{ female})$

Multiple-Step Profitability Analysis (Escrow / Management, Escrow / Management)

for the Nucleus

*not = Hitler-Zeuge Party (Wertheim, Numchelens)*

12 2000

## Definitions

`soft = Vether (numAgentsMin, numAgentsMax)`

~~Fonte = fonsuji~~

```
{ case 1: (if (is-ancestor? 38 (left))  
{
```

(0126) 424ms

Colorado 45 VWD

## Mémo Parcours (A)

Inhaler for many occasions, which ever you prefer.

691 Cuvier

WILHELM KARL MÜLLER

for the 1950's fut.

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartons tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes parties com vulguin i caldrà portar el compte de quantes parties ha guanyat cadascú.

**Exercici 1 (1 punt)**

Fer un procediment anomenat `CrearCartro` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1<sup>er</sup> a 15,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.
- 5 nombres del 31 al 45,

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

**UTILITZEU** (no implementar) els següents procediments o funcions:

- `int GenerarNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordene l'array a de dimensió dim de menor a major.

$$7 \times 3 = 213$$

$$213 + 328 = \underline{\underline{541}}$$

Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `True` en el cas que siguin iguals i `False` en cas contrari.

**NOTA:** Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$90 - 8 = 82$$

$$\begin{aligned} 82 + 43 &= 125 \\ \text{Resultat: } 125 \end{aligned}$$

**Exercici 3 (1 punt)**

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`index`) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

**NOTA.** Recordeu que els números de les boles van de 1 a 75.

**Exercici 4 (1 punt)**  
Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contindrà informació dels noms de les boles que han sortit. En l'array bombo si el valor d'una determinada posició (index) és un 1, vindrà dir que la bola corresponent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

La funció retornarà `True` en el cas que s'hagi aconseguit Bingo i `False` en cas contrari.

**Exercici 5 (1 punt)**  
Fer un procediment anomenat ImprimirComentari per imprimir un comentari sobre el marcador. El procediment rebrà dos valors enters que seran el nombre de partides guanyades per cada jugador. En el cas que la diferència sigui:

- menor o igual que 1, escriure el missatge "Està molt igualat".
- entre 2 i 5, el missatge hauria de ser "La remuntada és possible".
- Major o igual que 6, el missatge haurà de ser "Avui és el teu dia de sort".

**NOTA:** podeu utilitzar la funció `abs()` per obtenir el valor absolut d'un nombre.

**Exercici 5 (1 punt)**

Fer un procediment anomenat InicialitzarArray per inicialitzar els elements d'un vector v. El procediment rebrà un vector v de dimensió dim i un enter min que serà el mínim que pot prendre cada element del vector.

1. Declarar (`on corresponds`) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (`on corresponds`) els arrays necessaris per representar els cartons de dos jugadors anomenats cartoPlayer1 i cartoPlayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment `MenuPrincipal()` i llegir l'opció escollida.

4. Si l'opció és 1, generar dos cartons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartons dels jugadors tot a 0 amb el procediment `InicialitzarArray`.
  - 4.2. Utilitzar el procediment `CrearCartro` per crear els cartons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartons són diferents utilitzant la funció `ArraysIguals`. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment `InicialitzarArray`.
  - 5.3. Generar un nombre per simular la bola amb la funció `GeneraNombre`.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció `ImprimirBolesSortides` per imprimir els noms de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció `Escrutini` per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
  6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment `ImprimirComentari`.
  7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa..."
  8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
  9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.

Mostra per pantalla un menú amb les opcions del codi: 1. Assignar cartons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
    Inicialitza el vector a de dimensió dim al valor v
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

**Exercici 6 (2,5 punts)**  
Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenuPrincipal()
{
    cout << "--- Menú Principal ---" << endl;
    cout << "1.- Asignar cartons " << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

```
int GeneraNombre(int min, int max, int a[], int dim)
    Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim
```

```
void InicialitzarArray(int a[], int dim, int v)
    Inicialitza el vector a de dimensió dim al valor v
```

```
int GeneraNombre()
    Genera un nombre aleatori entre 0 i 100
```

```
bool ArraysIguals(int a[], int b[])
    Comprova si els dos arrays són iguals
```

```
int CrearCartro()
    Crea un cartó de bingo
```

```
void ImprimirBolesSortides(int a[])
    Imprimeix les boles que han sortit
```

```
void ImprimirComentari(int a[])
    Imprimeix el comentari del marcador
```

```
int Escrutini(int a[], int b[])
    Comprova si un cartó ha fet bingo
```



Exercici 7 (0.5 punts)  
Declarar un nou tipus de dades, TCartro, com un registre amb els camps: Ident\_Cartro (valor ente), Numeros (array de 25 enters) i Nom\_Jugador (cadena de 50 caràcters).

Fer un procediment anomenat Imprimir que rebi com a paràmetre un registre del tipus TCartro i imprimeix l'identificador del cartó i el nom del jugador separant -los amb el caràcter '-'.

Exercici 9 (1 punt)  
Escriviu la funció Elevar que retorna el resultat d'elevat un nombre al quadrat. La funció tira paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realitzar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclòs en l'interval [-46.340,46.340] per evitar overflow.

# Examen Parcial (23 de Novembre de 2020)

Nom estudiant:

Grup:

NIU:

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat. (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bumbo. Els cartells tenen un total de 25 números. En aquest tipus de bingo, es van treient boles del bumbo i es guanya un premi quan han sortit tots els números del nostre cartell. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

## Exercici 1 (1 punt)

Fer un procediment anomenat ClearCartell per a generar un cartell. S'han de generar 25 nombres del 1 al 75. Sha d'assegurar que hi hagin:

- 5 nombres del l'1 al 15.
- 5 nombres del 16 al 30.
- 5 nombres del 31 al 45.
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

Ei procediment rebrà un array (que serà el cartell) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- int GenerarNombre (int min, int max, int a[], int dim)
- void Ordena (int a[], int dim)

Ordena l'array a de dimensió dim de menor a major.

$$4 \times 5 = 20$$

$$20 + 4a = \boxed{69}$$

$$\begin{aligned} 8 \times 8 &= 64 \\ 64 - 18 &= 46 \\ \text{Resultat: } & 46 \end{aligned}$$

## Exercici 3 (1 punt)

Fer un procediment anomenat ImprimirBolesSortides per imprimir els números de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

NOTA. Recordeu que els números de les boles van de 1 a 75.

Fer la funció anomenada ArraysIguals per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà true en el cas que siguin iguals i false en cas contrari.

Fer una funció anomenada Escrutini per saber si un cartó ha aconseguit Bingo. La funció rebrà un array que serà el cartó, la dimensió del cartó i un array (bombo) que contindrà informació dels noms que han sortit. En l'array bombó si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola correspondent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

La funció retornarà True en el cas que s'hagi aconseguit Bingo i False en cas contrari.

Exercici 4 (1 punt)  
Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenuPrincipal()
{
    cout << "---- Menu Principal ----" << endl;
    cout << "1.- Asignar cartones" << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Asignar cartons, 2. Jugar, etc.

```
void InicializarArray(int a[], int dim, int v)
{
    int GenerarNombre(int min, int max, int a[], int dim);
    inicializa el vector a de dimensió dim al valor v
}
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:  
1. Declara (*on corresponguï*) les constants necessàries per tal que el programa sigui fàcilment modificable.

2. Declarar (*on corresponguï*) els arrays necessaris per representar els cartons de dos jugadors marcadors. En el cas que la diferència sigui:

- menor o igual que 1, escriure el missatge "Està molt igualat".
  - entre 2 i 5, el missatge hauria de ser "La remuntada és possible".
  - Major o igual que 6, el missatge haurà de ser "Avui és el teu dia de sort".
- NOTA: podeu utilitzar la funció abs() per obtenir el valor absolut d'un nombre.

Fer un procediment anomenat ImprimirComentari per imprimir un comentari sobre el marcador. El procediment rebrà dos valors enters que seran el nombre de partides guanyades per cada jugador. En el cas que la diferència sigui:

- menor o igual que 1, escriure el missatge "Està molt igualat".
- entre 2 i 5, el missatge hauria de ser "La remuntada és possible".
- Major o igual que 6, el missatge haurà de ser "Avui és el teu dia de sort".

- NOTA: podeu utilitzar la funció abs() per obtenir el valor absolut d'un nombre.
- Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:  
1. Declara (*on corresponguï*) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (*on corresponguï*) els arrays necessaris per representar els cartons de dos jugadors marcadors cartoPlayer1 i cartoPlayer2. Declarar també un array anomenat bombo, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment MenuPrincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartons (un per a cada jugador) següint els passos següents:  
4.1. Inicialitzar els cartons dels jugadors tot a 0 amb el procediment InicializarArray.  
4.2. Utilitzar el procediment CrearCarto per crear els cartons del jugador 1 i del jugador 2.
- 4.3. Comprovar si els dos cartons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:  
5.1. S'haura de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
- 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicializarArray.
- 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
- 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els nombres de les boles que han sortit al llarg de la partida.
- 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
- 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa..."
8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



Declarar un nou tipus de dades, TCarro, com un registre amb els camps: Ident\_Carro (valor enter), Numeros (array de 25 enters) i Nom\_Jugador (cadena de 50 caràcters).  
Fer un procediment anomenat Imprimir que rebi com a paràmetre un registre del tipus TCarro i imprimeix l'identificador del carro i el nom del jugador separant-los amb el caràcter ','.

Exercici 9 (1 punt)  
Escriviu la funció Elevar que retomí el resultat d'elevat un nombre al quadrat. La funció tindrà dos paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realitzar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'interval [-46.340,46.340] per evitar overflow.

### Exercici 8 (1 punt)

Fer una funció anomenada Es\_MatPositiva que rebí com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartons tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

#### Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartó` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

`UTILITZEU` (no implementar) els següents procediments o funcions:

- `int GenerarNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordena l'array a de dimensió dim de menor a major.

$$40 \times 20 = 800$$
$$100 - 50 = \underline{50}$$

$$12 \times 2 = 24$$
$$80 + 15 = 95$$

Resultado : 95

#### Exercici 3 (1 punt)

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingirà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (índex) és un 1, podrà dir que la bola corresponent ha sortit i si el és un 0, podrà dir que aquella bola no ha sortit.

NOTA: Recordeu que els números de les boles van de 1 a 75.

Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retorna `true` en el cas que siguin iguals i `false` en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

Fer una funció anomenada Escrutini per saber si un cartó ha aconseguit Bingo. La funció rebrà un array que serà el cartó, la dimensió del cartó i un array (bombo) que contingà informació dels nombres que han sortit. En l'array bombó si el valor d'una determinada posició (index) és un 1, vindrà dir que la bola correspondent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i False en cas contrari.

**Ejercici 6 (2,5 punts)**  
Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenuPrincipal()
{
    cout << "-- Menu Principal ---" << endl;
    cout << "1.- Asignar cartones" << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Asignar cartons, 2. Jugar, etc.

```
void InicializarArray(int a[], int dim, int v)
{
    int GenerarNombre(int min, int max, int a[], int dim)
    {
        // Implementació de la funció GenerarNombre
    }
}
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main()) que segueixi els següents passos:

- Declarar (on corresponguï) les constants necessàries per tal que el programa sigui fàcilment modificable.
- Declarar (on corresponguï) els arrays necessaris per representar els cartons de dos jugadors anomenats cartoPlayer1 i cartoPlayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit, quins no.
- Utilitzar el procediment MenuPrincipal() i llegir l'opció escollida.
- Si l'opció és 1, generar dos cartons (un per a cada jugador) seguint els passos següents:
  - Initialitzar els cartons dels jugadors tot a 0 amb el procediment InicializarArray.
  - Utilitzar el procediment CrearCartro per crear els cartons del jugador 1 i del jugador 2.
  - Comprovar si els dos cartons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
- Si l'opció és 2, implementa el joc seguint els passos següents:
  - Shaurà de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - Inicialitzar l'array bombó a 0 amb el procediment InicializarArray.
  - Generar un nombre per simular la bola amb la funció GenerarNombre.
  - Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els nombres de les boles que han sortit a llarg de la partida.
  - Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - En cas que no hi hagi cap guanyador tornar al punt 5.3
  - Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
  - Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa..."
  - Qualsevol altra opció, escriure el missatge: "Opció no permesa".
  - Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



Declarar un nou tipus de dades, TCarro, com un registre amb els camps: Ident\_Carro (valor enter), Numeros (array de 25 enters), i Nom\_Jugador (cadena de 50 caràcters).

Fer un procediment anomenat Imprimir que rebi com a parametre un registre del tipus TCarro i imprimeix l'identificador del carro i el nom del jugador separant-los amb el caràcter ','.

Exercici y (1 punt)

Escriviu la funció Elevar que retorni el resultat d'elevat un nombre al quadrat. La funció tindrà paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realitzar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'interval [-46.340, 46.340] per evitar overflow.

### Exercici 8 (1 punt)

Fer una funció anomenada Es\_MatPositiva que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de 1 al 75 dins del bumbo. Els cartons tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bumbo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caindrà portar el compte de quantes partides ha guanyat cadascú.

**Exercici 1 (1 punt)**

Fer un procediment anomenat `CrearCartro` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 41 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

**UTILITZEU** (no implementar) els següents procediments o funcions:

- `int GenerarNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min,max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordena l'array a de dimensió dim de menor a major.

$$100 \times 20 = \underline{\underline{2000}}$$
$$90 + 15 = \underline{\underline{105}}$$

$$12 + 10 = \underline{\underline{114}}$$
$$100 - 10 = \underline{\underline{90}}$$

**Resultados:** 90

Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `True` en el cas que siguin iguals i `False` en cas contrari.

**NOTA:** Podeu suposar que els dos arrays tenen la mateixa dimensió.

**Exercici 3 (1 punt)**

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contindrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`índex`) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

**NOTA:** Recordeu que els números de les boles van de 1 a 75.

Fer una funció anomenada `Escrutini` per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombó) que contingrà informació dels noms de les botes que han sortit. En l'array bombó si el valor d'una determinada posició (`index`) és un 1, vindrà dir que la bola correspondent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

La funció retornarà `True` en el cas que s'hagi aconseguit Bingo i `False` en cas contrari.

**Exercici 5 (2.5 punts)**

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void InicialitzarArray(int a[], int dim, int v)
{
    cout << " --- Menu Principal --- " << endl;
    cout << "1.- Asignar cartrons " << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Asignar cartrons 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
{
    cout << " --- Menu Principal --- " << endl;
    cout << "1.- Asignar cartrons " << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

int GenerarNombre(int min, int max, int a[], int dim)

Retorna un enter aleatori entre `min` i `max`, controlant que no estigui a l'array `a` de dimensió `dim`.

Fer un programa complet (declaracions globals i funció `main()`) que segueixi els següents passos:

1. Declarar (`on correspongui`) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (`on correspongui`) els arrays necessaris per representar els cartrons de dos jugadors anomenats `cartróPlayer1` i `cartróPlayer2`. Declarar també un array, anomenat `bombo`, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment `MenuPrincipal()` i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartrons dels jugadors tot a 0 amb el procediment `InicialitzarArray`.
  - 4.2. Utilitzar el procediment `CrearCartró` per crear els cartrons del jugador 1 i del jugador 2.
- 4.3. Comprovar si els dos cartrons són diferents utilitzant la funció `ArraysIguals`. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartrons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array `bombo` a 0 amb el procediment `InicialitzarArray`.
  - 5.3. Generar un nombre per simular la bola amb la funció `GenerarNombre`.
- 5.4. Imprimir el nombre que ha sortit i utilitzar la funció `ImprimirBolesSortides` per imprimir els noms de les botes que han sortit al llarg de la partida.
- 5.5. Utilitzar la funció `Escrutini` per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
- 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment `ImprimirComentari`.
7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa..."
8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0,5 punts)**

Declarar un nou tipus de dades, TCartro, com un registre amb els camps: Ident\_Cartro (valor enter), Numeros (array de 25 enters) i Nom\_Jugador (cadena de 50 caràcters).

Fer un procediment anomenat Imprimir que rebi com a paràmetre un registre del tipus TCartro i imprimeixi l'identificador del cartó i el nom del jugador separant-los amb el caràcter '-'.

**Exercici 9 (1 punt)**

Escrue la funció Elevar que retorni el resultat de elevar un nombre al quadrat. La funció tindrà paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si s'ha pogut realizar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'intervall [-46.340,46.340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada Es\_MatPositiva que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

1497995

cin &gt;&gt; id;

bool troubt != false;

int i;

while (i &lt; NCLIENT &amp;&amp; troubt)

```

if (troubt)
    {
        if (troubt)
            {
                else
                    }
            }
    }
}

```

if (troubt)

if (vcid == 0)

return 1;

else return 0;

&amp; troubt

cin &gt;&gt; id;

$$x = \frac{-b}{a}, b \neq 0$$

int x &gt;

0 &amp; f(b) &gt; 0;

if (a == 0)

return x == 0;

if (b &gt; 0)

3x = 0

$$x = -b/a$$

$$b = 0$$

$$1 = 0$$



卷之三

Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retorna `true` en el cas que siguin iguals i `false` en cas contrari.

**Important:** Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa; per tant, ha d'haver coherència entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el tenuu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartells tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartell. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadaescú.

$$100 + 83 = 183$$

$$21 \times 15 = 315$$

Resultat: 315

Fer un procediment anomenat `CrearCartiro` per a generar un cartíó. S'han de generar 25 noms del 1 al 75. Sha d'assegurar que hi hagin:

- 5 noms del 1 al 15
- 5 noms del 16 al 30
- 5 noms del 31 al 45
- 5 noms del 46 al 60
- 5 noms del 61 al 75

- 5 nombres del 11 al 15.
  - 5 nombres del 16 al 30.
  - 5 nombres del 31 al 45.
  - 5 nombres del 46 al 60.
  - 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartró) i la dimensió de l'array. Retornarà els números ordenats de menor a major.

El procediment rebrà un array (que serà el cartell) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

- int GenerarNombre (int min, int max, int a[], int dim)
 

Genera un nombre aleatori entre min i max. I revisa que no estigui a l'array a de dimensió dim.
  - void Ordena (int a[], int dim)

Ordena l'array a de dimensió dim de menor a major.

$$20 + 40 = 60$$

$$80 \times 2 =$$

**Exercici 3 (1 punt)**  
Fer un procediment anomenat ImprimirBolesSortides per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (*index*) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0 voldrà dir que aquella bola no ha sortit.

#### Exercici 4 (1 punt)

Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contindrà informació dels nombres que han sortit. En l'array bombó si el valor d'una determinada posició (index) és un 1, vindrà dir que la bola correspondent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

#### Exercici 6 (2,5 punts)

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenuPrincipal ()  
{  
    cout << " --- Menu Principal --- " << endl;  
    cout << "1.- Asignar cartrons " << endl;  
    cout << "2.- Jugar" << endl;  
    cout << "3.- Marcador" << endl;  
    cout << "4.- Sortir" << endl;  
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Asignar cartrons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)  
Iniciaitza el vector a de dimensió dim al valor v
```

int GenerarNombre (int min, int max, int a[], int dim)  
Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartons de dos jugadors anomenats cartroPlayer1 i cartroPlayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment MenuPrincipal () i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CrearCartro per crear els cartons del jugador 1 i del jugador 2.
- 4.3. Comprovar si els dos cartons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc següent els passos següents:
  - 5.1. Shaurà de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els nombres de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
  6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
  7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa . . .".
  8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
  9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0,5 punts)**  
Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `Nom_Jugador` (cadena de 50 caràcters).

Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeixi l'identificador del cartó i el nom del jugador separant-los amb el caràcter '-'.

**Exercici 9 (1 punt)**  
Escrive la funció `Elevar` que retorni el resultat de elevar un nombre al quadrat. La funció tindrà dos paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si s'ha pogut realizar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha de estar inclos en l'interval [-46.340,46.340] per evitar overflow.

### Exercici 8 (1 punt)

Fer una funció anomenada `Esp_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorni 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

14/10/88

do

| void NewPrincipal();

in =&gt; opas;

switch (opas);

case 1: do

| cout &lt;&lt; "Introducir el ID del identificador";

cin =&gt; userObject;

| while (nucleo &lt; 0 || n &gt; max) n = client &gt;&gt; clients;

| voter Votar (MIN, MAX) ((declaracion-hoja));

| votaciones[i] = VOTL[i];

|

| break;

| cout &lt;&lt; "Res: " &lt;&lt; res &lt;&lt; endl;

case 2: while (nucleo &amp;amp; n &gt;= 0) clients

| res = MetodoZero (votaciones, clients);

| if (res == 1) ticket = true;

| cout &lt;&lt; "Ticket " &lt;&lt; res &lt;&lt; endl;

else

| InicializarHoja (escritor, ABENTS, 0);

| RecuperarVotaciones, CLIENTS, escritor);

| for (int i = 0; i &lt; hoja.S; i++)

cout &lt;&lt; "Escritor[" &lt;&lt; i &lt;&lt; endl;

break;

case 3: if (res == 1)

| cout &lt;&lt; "Ticket = false";

| while (nucleo &amp;amp; n &gt;= 0) ticket)

| res = MetodoZero (escritor, ABENTS);

| if (res == 1)

| cout &lt;&lt; "Ticket no es falso" &lt;&lt; endl;

| break;

$b = i \sqrt{m}$

hence

if  $\psi$  is a solution of Schrödinger's equation then

hence

$\psi = u \psi_0$  where  $\psi_0$  is a solution of

$\frac{d^2\psi}{dx^2} + k^2 \psi = 0$  with boundary conditions  $\psi(0) = 0$  and  $\psi(L) = 0$

**Important:** Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.).

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartons tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

**Exercici 1 (1 punt)**  
Fer un procediment anomenat CrearCartro per a generar un cartó. S'han de generar 25 números del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1º al 15.
- 5 nombres del 16 al 30.
- 5 nombres del 31 al 45.
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- int GenerarNombre (int min, int max, int a[], int dim)
  - Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- void Ordena (int a[], int dim)
  - Ordena l'array a de dimensió dim de menor a major.

$$22 \times 53 = \underline{423}$$

$$423 - 7 = \underline{121}$$

Fer la funció anomenada ArraysIguals per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà True en el cas que siguin iguals i False en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$14 + 6 = 20$$

$$20 - 2 = 18$$

$$\text{Resultat} = 18$$

**Exercici 3 (1 punt)**  
Fer un procediment anomenat ImprimirBolesSortides per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingirà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

NOTA: Recordeu que els números de les boles van de 1 a 75.

**Exercici 4 (1 punt)**

Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contindrà informació dels nombres que han sortit. En l'array bombó si el valor d'una determinada posició (index) és un 1, vindrà dir que la bola correspondent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

**Exercici 6 (2,5 punts)**

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void InicialitzarArray(int a[], int dim, int v)
{
    cout << "1. - Menu Principal ---" << endl;
    cout << "2. - Jugar" << endl;
    cout << "3. - Marcador" << endl;
    cout << "4. - Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Assignar cartrons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
{
    int GenerarNombre(int min, int max, int a[], int dim)
    inicialitza el vector a de dimensió dim al valor v
}
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim.

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartrons de dos jugadors anomenats cartróPlayer1 i cartróPlayer2. Declarar també un array, anomenat bombo, per controlar quins noms han sortit i quins no.
3. Utilitzar el procediment MenuPrincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartrons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CrearCartró per crear els cartrons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartrons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartrons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els noms de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
  6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
  7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa...".
  8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
  9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



Exercici 7 (0,5 punts)

Declarar un nou tipus de dades, `TCarro`, com un registre amb els camps: `Ident_Carro` (valor enter), `Numeros` (array de 25 enters) i `nom_Jugador` (cadena de 50 caràcters).

Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCarro` i imprimeixi l'identificador del carro i el nom del jugador separant-los amb el caràcter '-'.

Exercici 8 (1 punt)

Escriure la funció `Elevar` que retorni el resultat d'elevat un nombre al quadrat. La funció tindrà paràmetres de tipus `int`. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realitzar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'intervall [-46.340,46.340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada `E8_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorni 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

⑥ int IndexMaximArray( int V2[], int dim ) {  
 1

int Posmax = 0, ind = 0;

for( int i = 0; i < dim; i++ ) {  
 2

if( V2[i] > Posmax ) {  
 3

Posmax = V2[i];

ind = i;

}  
 3

}  
 2

return ind;

}  
 1

{

144

1400 = true

return 0

{ else }

true = false

return 1

{ } = C(LN) f!

while ( ! ( a < m ) || ( ! ( a > r ) ) ) {

base + base = jbase

i = i + 1;

} ( w + v ) / m + d ) w

⑥

w < w

cout << "Enter value for the integer" ;

while ( ( w < w ) || ( w > w ) )

⑦

**Important:** Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartrons tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

#### Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartro` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 31 al 45,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

**UTILITZEU (no implementar) els següents procediments o funcions:**

- `int GenerarNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordena l'array a de dimensió dim de menor a major.

#### Exercici 2 (1 punt)

Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `true` en el cas que siguin iguals i `false` en cas contrari.

**NOTA:** Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$4 \times 3 = 12$$

$$12 \times 2 = 36$$

Resolució: 36

#### Exercici 3 (1 punt)

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contendrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`index`) és un 1, vindrà dir que la bola corresponent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

**NOTA:** Recordeu que els números de les boles van de 1 a 75.

**Exercici 4 (1 punt)**  
Fer una funció anomenada Escrutini per saber si un cartó ha aconseguit Bingo. La funció rebrà un array que serà el cartó, la dimensió del cartó i un array (bombo) que contindrà informació dels nombres que han sortit. En l'array bombo si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

La funció retornarà True en el cas que s'hagi aconseguit Bingo i False en cas contrari.

**Exercici 5 (2.5 punts)**  
Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenuPrincipal()
{
    cout << "---- Menu Principal ----" << endl;
    cout << "1.- Asignar cartons" << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Mostra per pantalla un menú amb les options del codi: 1. Assignar cartons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
Inicialitza el vector a de dimensió dim al valor v
```

```
int GenerarNombre (int min, int max, int a[], int dim)
Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim
```

Fer un programa complet (declaracions globals i funció main()) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartons de dos jugadors anomenats cartoPlayer1 i cartoPlayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment MenuPrincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CreaCarto per crear els cartons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. Shaurà de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els nombres de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa...".
8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0.5 punts)**

Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `Nom_Jugador` (cadena de 50 caràcters).

Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeixi l'identificador del cartro i el nom del jugador separant-los amb el caràcter ‘,’.

**Exercici 8 (1 punt)**

**Exercici 9 (1 punt)**

Escriure la funció `Elevar` que retorna el resultat d'elevat un nombre al quadrat. La funció tindrà dos paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realitzar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

**NOTA:** El valor de la base ha d'estar inclòs en l'intervall [-46.340,46.340] per evitar overflow.

1. 96 advances

14923(9)

all define  $\text{My}_n$  class & b

int

shape def & init

MyVector & MyMatrix

decomp & init

int & init

int & init

{ Viz.

int & void func [My & class]

for (int i = 0; i < N; i++)

int & void func (int i, int j, int k)

2. ~~for (int i = 0; i < N; i++)~~

int & void func (int i, int j)

book (class) + = symb;

not note

while (loop condition)

    do something

    if (condition)

        break

else

    if (condition)

        continue

else

    execute the code

    return value

    exit loop

    for ( )

        i = i + 1

    do ( )

7.  $\omega_0$

$$\omega_0/q = \pm x$$

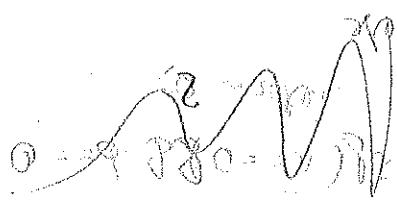
8.  $\omega_0$

$\omega_0$

9.  $\omega_0$

$$(\omega = \omega_0)^0$$

$$(\omega = \omega_0)^2$$



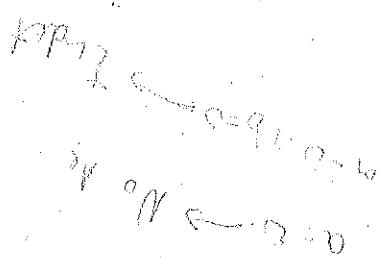
~~( $\omega = \omega_0$ )  $\rightarrow$   $\omega_0^2$~~

$\omega_0$

$\omega_0$

10.  $\omega_0$

$$(\omega = \omega_0)^2$$



No. 12 solution

$$a_1 = 0 \quad \left( \frac{3}{q} - \infty \right) \text{ or } a_1 = 0$$

#include <iostream> (499509)  
#include ...  
#define DIM 12  
#define DMM 150  
vs  
using namespace std;  
int main()  
{ int vots, votclients, votcomercials;  
int clients;  
int Votacions[ ];  
int Escrutini[ ];

void Votar (int min, int max){  
 int a;  
 cin >> a;  
 do{  
 if (a < min & a >= max){  
 cout << "Vot estres correctament";  
 } else  
 cout << "Valor fora de  
 l'interval";  
 } while (a <= min & a >= max);

~~definir~~  
if (array1[i] = 0)  
for (int i = 0; i > 0; i++)  
 if array1[i] = 0  
 if (array1[i] = 0)  
 continue return 1;  
 else  
 return 0;

80  
  
int IndexMaximArray (array1[], array2[]){

8



Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `true` en el cas que siguin iguals i `false` en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el tenuiu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartrons tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

#### Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartó` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 11 al 15,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementeu) els següents procediments o funcions:

- `int GenerarNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordene l'array a de dimensió dim de menor a major.

$$20 \times 10 = 200$$
$$200 + 50 = 250$$

#### Exercici 4 (1 punt)

Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contingrà informació dels noms de les boles que han sortit. En l'array bombó si el valor d'una determinada posició (índex) és un 1, vindrà dir que la bola correspondent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

#### Exercici 6 (2.5 punts)

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenuPrincipal()
{
    cout << "---- Menu Principal ----" << endl;
    cout << "1. - Asignar cartrons" << endl;
    cout << "2. - Jugar" << endl;
    cout << "3. - Marador" << endl;
    cout << "4. - Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Assignar cartrons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
Inicialitza el vector a de dimensió dim al valor v
```

```
int GenerarNombre (int min, int max, int a[], int dim)
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartrons de dos jugadors anomenats cartroPlayer1 i cartroPlayer2. Declarar també un array, anomenat bombo, per controlar quins noms han sortit i quins no.
3. Utilitzar el procediment MenuPrincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartrons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CrearCartro per crear els cartrons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartrons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartrons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els noms de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
  6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
  7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa..."
  8. Qualsevol altra opció, escriure el missatge: "Opció no permesa."
  9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0,5 punts)**  
Declarar un nou tipus de dades, TCartro, com un registre amb els camps: Ident\_Cartro (valor enter), Numeros (array de 25 enters) i Nom\_Jugador (cadena de 50 caràcters).

Fer un procediment anomenat Imprimir que rebi com a paràmetre un registre del tipus TCartro i imprimeix l'identificador del cartró i el nom del jugador separant-los amb el caràcter '-'.

**Exercici 9 (1 punt)**  
Escriure la funció Elevar que retorna el resultat de elevar un nombre al quadrat. La funció tindrà paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

**NOTA:** El valor de la base ha d'estar inclos en l'intervall [-46.340, 46.340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada Es\_MatPositiu va que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

$$ax + b = 0$$

$$\frac{-b}{x} = a$$

case 1:

$$(m >> 1den);$$

~~while (iden > (849) & & (iden < 0))~~

$$(m >> 1den);$$

$$ax = -b$$

$$-b/a = x$$



for ( $i = 0$ ;  $i < \dim$ ;  $i++$ )

{

$\text{if } (\text{vote}[i] == \emptyset)$

$(\text{vote}[i] = \{1\})$

$\text{vote}[i].add(2)$

$\text{vote}[i].add(3)$

}

$\text{else if } (\text{vote}[i] > \text{max}[i])$

3

2

int  $\delta = 101$ ;

104

112

101

for ( $i = 0$ ;  $i < \dim$ ;  $i++$ )

{

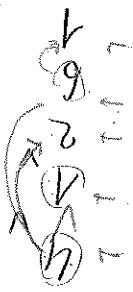
do

{

$\text{Escutini}[\text{vote}[i]]++$ ;

if ( $\text{vote}[i] == \delta$ )  $\text{Obra}.$

8



return veto

exit CC

while ( $min \leq max$ )

$min = max$

if ( $min > max$ ) Euro

do ~~but Voter~~ <sup>do</sup> ~~not Voter~~, <sup>do</sup> ~~and min, and max~~

End

\* define N-CLIENTS 150

\* define N-ACON 12

O - 149 To be voted

Vofato 150 clients

12 aquiles computers  $\rightarrow$  will be connected

+ order objective 111-112

Hinrichsd (vstavan)  
H define Client & SO

14013488

H  
int main () {  
int Voteriom [Clients] ; int opisó;  
int Ercetion [Agents] ; int idetification;  
InvalitforAvryg (Clients, Voteriom [0])



Meni Principal()  
cin >> opisó;  
Switch (opisó) {  
Case "1":

Whille {

Do  
if (opisó == 1) {  
cin >> opisó;

#define COUNT 12

#define COUNT 15  
int sorted[COUNT];  
int solution[COUNT];

while ( $a < \min \{ i, o - \text{rank} \}$ ) {

    cout << "middle:  
    cout << "left:  
    cout >> a;

    j

for (i = 0; i < Dim; i++)  
    cout << solution[i] - minid) + ";

int i=0, a = array[0], min; j  
for (i=0; i < Dim; i++)  
    if (array[i] > a)  
        min = array[i];  
    a = array[i];

**Examen Parcial (23 de Novembre de 2020)****Nom estudiant:****Grup: 45**  
**NIU: 4055291**

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherència entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

E! Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartells tenen un total de 25 números. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els números del nostre cartell. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

**Exercici 1 (1 punt)**

Fer un procediment anomenat `CreateCartro` per a generar un cartell. S'han de generar 25 nombres del 1 al 75. Si ha d'assegurar que hi hagin:

- 5 nombres del 1 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 31 al 45,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartell) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- `int GenerarNombre(int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordena l'array a de dimensió dim de menor a major.

$$20 \times 10 = 200$$

$$200 + 50 = \underline{\underline{250}}$$

**Exercici 2 (1 punt)**  
Fer a funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retorna `True` en el cas que siguin iguals i `False` en cas contrari.

**NOTA:** Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$4 \times 3 = 10$$

$$10 \times 2 = 20$$

Rosellot: 20

**Exercici 3 (1 punt)**

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contindrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`index`) és un 1, vendrà dir que la bola corresponent ha sortit i si el és un 0, vendrà dir que aquella bola no ha sortit.

**NOTA:** Recordeu que els números de les boles van de 1 a 75.

#### Exercici 4 (1 punt)

Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contindrà informació dels nombres que han sortit. En l'array bombo si el valor d'una determinada posició (index) és un 1, vindrà dir que la bola corresponent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

#### Exercici 5 (2,5 punts)

Fer una funció anomenada ImprimirComentari per imprimir un comentari sobre el marcador. El procediment rebrà dos valors enters que seran el nombre de partides guanyades per cada jugador. En el cas que la diferència sigui:

- menor o igual que 1, escriure el missatge "Està molt igualat".
- entre 2 i 5, el missatge hauria de ser "La remuntada és possible".
- Major o igual que 6, el missatge haurà de ser "Avui és el teu dia de sort".

NOTA: podeu utilitzar la funció abs() per obtenir el valor absolut d'un nombre.

#### Exercici 5 (1 punt)

Fer un procediment anomenat ImprimirComentari per imprimir un comentari sobre el marcador. El procediment rebrà dos valors enters que seran el nombre de partides guanyades per cada jugador. En el cas que la diferència sigui:

- menor o igual que 1, escriure el missatge "Està molt igualat".
  - entre 2 i 5, el missatge hauria de ser "La remuntada és possible".
  - Major o igual que 6, el missatge haurà de ser "Avui és el teu dia de sort".
- NOTA: podeu utilitzar la funció abs() per obtenir el valor absolut d'un nombre.
- Exercici 5 (1 punt)**  
Fer un programa complet (declaracions globals i funció main()) que segueixi els següents passos:
1. Declarar (on corresponguí) les constants necessàries per tal que el programa sigui fàcilment modificable.
  2. Declarar (on corresponguí) els arrays necessaris per representar els cartrons de dos jugadors anomenats cartroPlayer1 i cartroPlayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.
  3. Utilitzar el procediment MenuPrincipal () i llegir l'opció escollida.
  4. Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
    - 4.1. Inicialitzar els cartrons dels jugadors tot a 0 amb el procediment InicialitzarArray.
    - 4.2. Utilitzar el procediment CrearCartro per crear els cartrons del jugador 1 i del jugador 2.
    - 4.3. Comprovar si els dos cartrons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
  5. Si l'opció és 2, implementa el joc seguint els passos següents:
    - 5.1. S'haurà de comprovar que s'han generat els cartrons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
    - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
    - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
    - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els noms de les boles que han sortit al llarg de la partida.
    - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
    - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
    6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
    7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa . . .".
    8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
    9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



Exercici 7 (0.5 punts)

Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `Nom_Jugador` (cadena de 50 caràcters).

Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeixi l'identificador del cartó i el nom del jugador separant-los amb el caràcter `'`.

Exercici 9 (1 punt)

Escrivre la funció `Elevar` que retorni el resultat elevar un nombre al quadrat. La funció tindrà paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'intervall [46,340,46,340] per evitar overflow.

Exercici 8 (1 punt)

Fer una funció anomenada `Es_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

int array [ ] , int sum , int value;

for ( i = 0 ; i < 100 ; i ++ )

{  
array [i] = random value \* 100

value = value / 100 ;  
if

for ( i = 0 ; i > 0 ; i -- )

while ( i < 100 & & !break )

i ++ ;

if ( array [i] <= 0 )  
break = true ;

Third example {  
int char  
valuetwo [ ] , int N , int Escalator [ ] }

if ( i > 0 )

for ( i = 0 ; i < N ; i++ , valuetwo [i] )

switch ( Escalator [i] )

case 1 : Escalator [i] ++ ;

case 2 : Escalator [i] ++ ;

case 3 : Escalator [i] ++ ;



VOTS
A
O
I
Q
A
A

Conversan
S
S
9

1493112

26nd Mar 21 2014

operación 1

operación 2

operación 3

$$\text{operación 1} = x_2 - x_1$$

operación 2 = float(sqrt((float operacion 1)<sup>2</sup>))

$$\text{operación 2} = y_2 - y_1$$

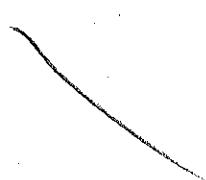
operación 2 = float(sqrt((float operacion 2)<sup>2</sup>))

if (operación 2 == operación 2)

return 0

else

}



0	1	4	8	1	5	7	8	1	0	1	2
---	---	---	---	---	---	---	---	---	---	---	---

Nom estudiant: Ched Meltwes

Exercici 1 (1 punt)

Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `True` en el cas que siguin iguals i `False` en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.) Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherència entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartells tenen un total de 25 números. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els números del nostre cartell. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartell` per a generar un cartell. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 31 al 45,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartell) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- `int GenerarNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre `min` i `max`, i revisa que no estigui a l'array a de dimensió `dim`.
- `void Ordena (int a[], int dim)`  
Ordena l'array a de dimensió `dim` de menor a major.

$$20 \times 10 = 200$$

$$200 + 50 = 250$$

$$\begin{aligned} & 4 \times 3 = 12 \\ \text{Resultat: } & 38 \end{aligned}$$

**Exercici 4 (1 punt)**  
Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contendrà informació dels nombres que han sortit. En l'array bombo si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

**Exercici 6 (2,5 punts)**  
Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenuPrincipal()
{
    cout << " --- Menu Principal --- " << endl;
    cout << " 1. - Asignar cartons " << endl;
    cout << " 2. - Jugar " << endl;
    cout << " 3. - Marcador " << endl;
    cout << " 4. - Sortir " << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Asignar cartons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
```

Inicialitza el vector a de dimensió dim al valor v

int GenerarNombre (int min, int max, int a[], int dim)
Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartons de dos jugadors anomenats cartróPlayer1 i cartróPlayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment MenuPrincipal () i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartons (un per a cada jugador) seguint els passos sequents:
  - 4.1. Inicialitzar els cartons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CrearCarto per crear els cartons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els nombres de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
  6. Si l'opció es 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
  7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa...".
  8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
  9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0,5 punts)**  
Declarar un nou tipus de dades, TCartro, com un registre amb els camps: Ident\_Cartro (valor enter), Numeros (array de 25 enters) i Nom\_Jugador (cadena de 50 caràcters).

Fer un procediment anomenat Imprimir que rebi com a paràmetre un registre del tipus TCartro i imprimeixi l'identificador del cartró i el nom deljugador separant-los amb el caràcter ‘:’.

**Exercici 9 (1 punt)**  
Escrivre la funció Elevar que retorni el resultat de elevar un nombre al quadrat. La funció tindrà dos paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si sha pogut realizar l'operació i un 0 en cas contrari.

**NOTA:** El valor de la base ha d'estar inclos en l'intervall [-46.340,46.340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada Es\_MatPositiva que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorni 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

int indexMinInArray (int array[], int dim)

```
{
    int i; minimum = array[0];
    int pos;
    for (i=0; i< dim; i++)
        if (array[i] < minimum)
            minimum = array[i];
            pos = i;
}
```

1497689

Vals [valores]

0	1	7	6	10	12	3	1	8	15	20
---	---	---	---	----	----	---	---	---	----	----

Popularit [concentración]

1	2	4	1	6	22	...	54	72	7	15
---	---	---	---	---	----	-----	----	----	---	----

int Valores (int min, int max)

```
{
    int valor;
```

(out => "Extra Valores entre xx: yy;

Am <= valor;

while (valor > max) || (valor < min)

}

cout >>

Void InitializeArray (int array[], int dim, int valor)

```
{
    int i;
```

for (i=0; i< dim; i++)

{

array[i] = valor;

}

int NoZeroArray (int array[], int dim)

```
{
    int i; retorno;
```

for (i=0; i< dim; i++)

{

while (array[i] == 0)

if (array[i] == 0)

```
{
        retorno = 0;
```

}

else

```
{
    retorno = 1;
```

}

Population =  $\sum_{i=1}^n V_i$

Population =  $\sum_{i=1}^n V_i$

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0.52	1.1	1.7	2.3	2.9	3.5	4.1	4.7	5.3	5.9	6.5	7.1	7.7	8.3	8.9	9.5

World population (without China, without India, without US)

Population =  $\sum_{i=1}^n V_i$

Population

$$S = 15 \quad Q = 1+1 = 2 \quad V = 15 \times 10^9$$

$$S = 3 \quad Q = 10^9$$

$$S = 3 \quad Q = 10^9$$

$$S = 0.1 \quad Q = 10^9$$

$$(Q = 10^9)$$

$$S = 1$$

Population (without China)

$$S = 15 \times 10^9$$

$$V =$$

$$V = 14 \times 10^9$$

$$V = 14 \times 10^9$$

Population (without India)

Population (without US)

$$Q = 10^9$$

$$S = 1$$

Population (without US)

$$S = 1$$

0.52	*	*	*	*
------	---	---	---	---

*	*	*	*	0
---	---	---	---	---

# Exercicis d'Informativa (103806)

Curs 2020-2021

Examen Parcial (23 de Novembre de 2020)

Grup:

NIU: 8565110

Nom estudiant: CASSIÉ SARDUAL

**Exercici 2 (1 punt)**  
Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics.  
La funció rebrà dos arrays i la seva dimensió. Retornarà `True` en el cas que siguin iguals i `False` en cas contrari.

**NOTA:** Podeu suposar que els dos arrays tenen la mateixa dimensió.

**Important:** Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.).

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'aixar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartons tenen un total de 25 números. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

**Exercici 1 (1 punt)**

Fer un procediment anomenat `CrearCartó` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1'1 al 15.
  - 5 nombres del 46 al 60 i
  - 5 nombres del 61 al 75.
  - 5 nombres del 31 al 45.
- El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- `int GenerarNúmero (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordene l'array a de dimensió dim de menor a major.

$$20 \times 20 = 400$$

$$7 \div 7 = \boxed{1}$$

**Exercici 3 (1 punt)**

Fer un procediment anomenat `ImprimirBolesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contindrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`index`) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

**NOTA.** Recordeu que els números de les boles van de 1 a 75.

$$300 + 300 = 600$$
$$150 \times 2 = 300$$

$$\text{RESUM} = 100$$

#### Exercici 4 (1 punt)

Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contingrà informació dels nombres que han sortit. En l'array bombo si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

**Exercici 6 (2.5 punts)**  
Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenuPrincipal()
{
    cout << "---- Menú Principal ----" << endl;
    cout << "1.- Asignar cartrons" << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Assignar cartrons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
{
    int GenerarNombre(int min, int max, int a[], int dim)
    {
        // Implementació de la funció GenerarNombre
    }
}
```

Inicialitza el vector a de dimensió dim al valor v

```
int InicialitzarArray(int a[], int dim, int v)
{
    for (int i = 0; i < dim; i++)
    {
        a[i] = v;
    }
}
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

```
int GenerarNombre(int min, int max, int a[], int dim)
{
    int numero = rand() % (max - min + 1) + min;
    while (numero == a[0] || numero == a[1] || numero == a[2])
    {
        numero = rand() % (max - min + 1) + min;
    }
    return numero;
}
```

ImprimeixComentari per imprimir un comentari sobre el marcador. El procediment rebrà dos valors enters que seran el nombre de partides guanyades per cada jugador. En el cas que la diferència sigui:

- menor o igual que 1, escriure el missatge "Està molt igual".
- entre 2 i 5, el missatge hauria de ser "La remunitada és possible".
- Maior o igual que 6, el missatge haurà de ser "Avui és el teu dia de sort".

NOTA: podeu utilitzar la funció abs() per obtenir el valor absolut d'un nombre.

#### Exercici 5 (1 punt)

Fer un procediment anomenat ImprimirComentari per imprimir un comentari sobre el marcador. El procediment rebrà dos valors enters que seran el nombre de partides guanyades per cada jugador. En el cas que la diferència sigui:

- menor o igual que 1, escriure el missatge "Està molt igual".
- entre 2 i 5, el missatge hauria de ser "La remunitada és possible".
- Maior o igual que 6, el missatge haurà de ser "Avui és el teu dia de sort".

NOTA: podeu utilitzar la funció abs() per obtenir el valor absolut d'un nombre.

- Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:
- Declarar (on corresponguï) les constants necessàries per tal que el programa sigui fàcilment modificable.
  - Declarar (on corresponguï) els arrays necessaris per representar els cartrons de dos jugadors anomenats cartroPlayer1 i cartroPlayer2. Declarar també un array, anomenat bombo, per controlar quins noms han sortit i quins no.
  - Utilitzar el procediment MenuPrincipal () i llegir l'opció escollida.
  - Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
    - Initialitzar els cartrons dels jugadors tot a 0 abix el procediment InicialitzarArray.
    - Utilitzar el procediment CrearCartro per crear els cartrons del jugador 1 i del jugador 2.
    - Comprovar si els dos cartrons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
  - Si l'opció és 2, implementa el joc seguint els passos següents:
    - S'haurà de comprovar que s'han generat els cartrons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
    - Initialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
    - Generar un nombre per simular la bola amb la funció GenerarNombre.
    - Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els noms de les boles que han sortit al llarg de la partida.
    - Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
    - En cas que no hi hagi cap guanyador tornar al punt 5.3
  - Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
  - Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa...".
  - Qualsevol altra opció, escriure el missatge: "Opció no permesa".
  - Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0,5 punts)**

Declarar un nou tipus de dades, TCartró, com un registre amb els camps: Ident\_Cartró (valor enter), Números (array de 25 enters) Nom\_Jugador (cadena de 50 caràcters). Fer un procediment anomenat Imprimir que rebi com a paràmetre un registre del tipus TCartró i imprimeix l'identificador del cartró i el nom del jugador separant-los amb el caràcter '-'.

**Exercici 9 (1 punt)**

Escriviu la funció Elevar que retorni el resultat d'elevat un nombre al quadrat. La funció tindrà 2 paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realitzar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

**NOTA:** El valor de la base ha d'estar inclos en l'intervall [-46.340,46.340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada Es\_MatPositiva que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

Vladyslav Sobol Niv: 1803148

```
#include <iostream>
using namespace std;

int main() {
    char m[12];
    Escrutini 12
    Notacions 9 150
    int i, vots;
    for (i=0; i<Escrutini; i++) {
        vots=Notacions[i]-101;
        Escrutini[not]++;
```

6.2.3 Escrutini +101;

a. do while int i;

```
    case '3':
        if (recompte==true)
            i = Index maxim Array (Escrutini, 12);
        cout << "El comercial guanyador es el cc i+104";
    else
        cout << "Encara no s'ha fet el recompte de vots.
```

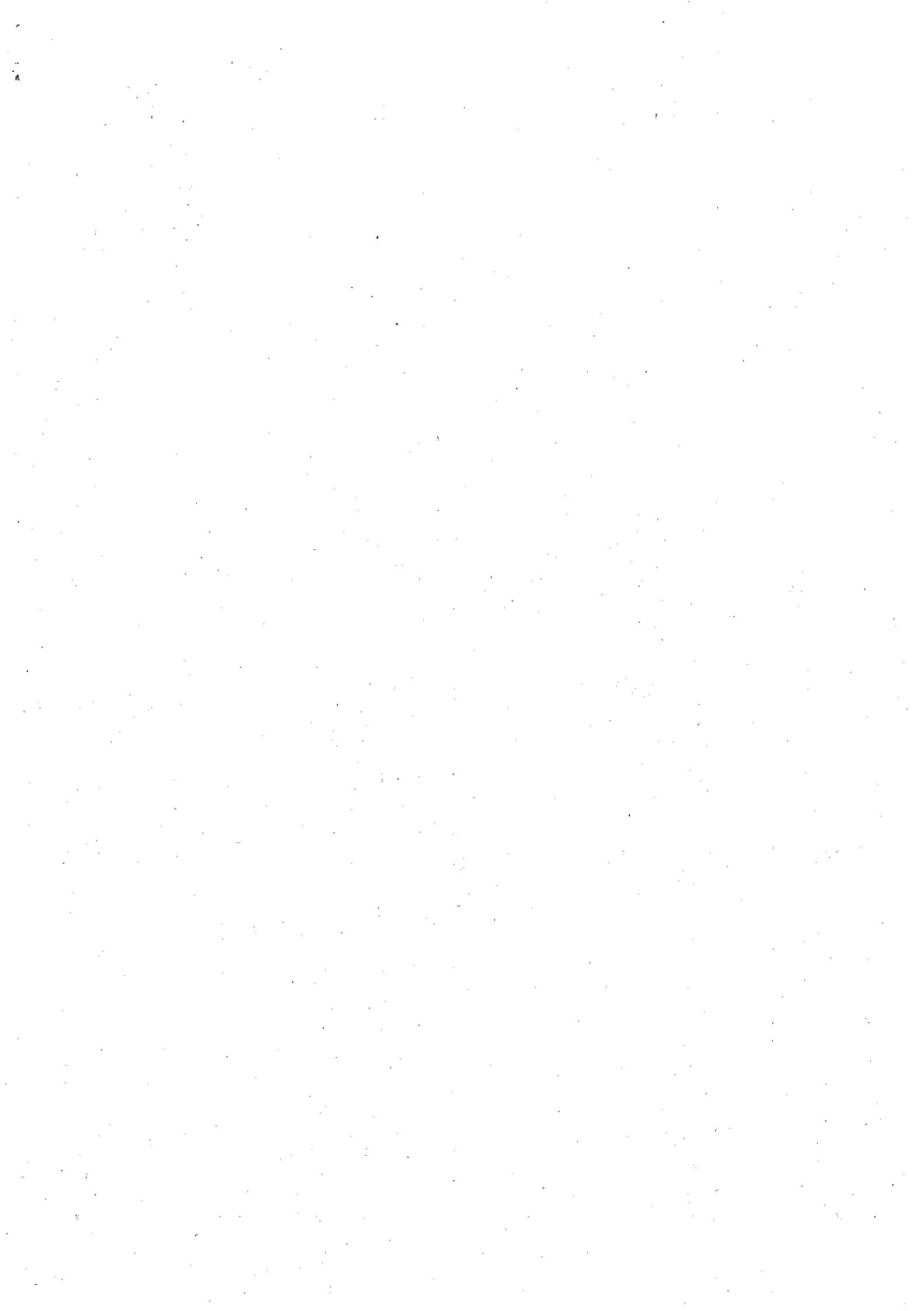
b = switch

```
    case '1':
        break;
    case '2':
        break;
    default:
        cout << "Opció no permet" / endf
```

```
    while (m!= '4');
    return 0;
```

8 & 8

$a=0 \rightarrow 0$   
 $b=0 \rightarrow 1$   
 $c:b=0 \rightarrow 00$



# Exercicis d'Informativa (103806)

## Examen Parcial (23 de Novembre de 2020)

Nom estudiant:

Curs 2020-2021  
Grup:  
NIU: 2426524

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartrons tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

Exercici 1 (1 punt)

Fer un procediment anomenat CrearCartó per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1'1 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 31 al 45,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- int GenerarNombre (int min, int max, int a[], int dim)  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- void Ordena (int a[], int dim)  
Ordena l'array a de dimensió dim de menor a major.

$$25 + 13 = \underline{38}$$
$$10 + 2 = \underline{12}$$

Exercici 2 (1 punt)

Fer la funció anomenada ArraysIguals per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà true en el cas que siguin iguals i false en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$6 \times 6 = 36$$
$$82 \times 4 = 336$$
$$\text{RESULTAT} = 12$$

Exercici 3 (1 punt)

Fer un procediment anomenat ImprimirBolesSortides per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (index) és un 1, vindrà dir que la bola corresponent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

NOTA: Recordeu que els números de les boles van de 1 a 75.

#### Exercici 4 (1 punt)

Fer una funció anomenada `Escrutini` per saber si un cartó ha aconseguit Bingo. La funció rebrà un array que serà el cartó, la dimensió del cartó i un array (bombo) que contingrà informació dels nombres que han sortit. En l'array del cartó si el valor d'una determinada posició (`index`) és un 1, vindrà dir que la bola correspondent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

La funció retornarà `true` en el cas que s'hagi aconseguit Bingo i `false` en cas contrari.

#### Exercici 5 (2,5 punts)

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenúPrincipal()
{
    cout << " --- Menú Principal --- " << endl;
    cout << "1.- Asignar cartones " << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Salir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Assignar cartons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
{
    int GeneraNombre(int min, int max, int a[], int dim)
    {
        //Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim
    }
}
```

Fer un programa complet (declaracions globals i funció main) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartons de dos jugadors anomenats `cartoPlayer1` i `cartoPlayer2`. Declarar també un array, anomenat `bombo`, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment `MenúPrincipal()` i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartons dels jugadors tot a 0 amb el procediment `InicialitzarArray`.
  - 4.2. Utilitzar el procediment `CrearCarto` per crear els cartons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartons són diferents utilitzant la funció `ArraysIguals`. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc següint els passos següents:
  - 5.1. Si haurà de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array `bombo` a 0 amb el procediment `InicialitzarArray`.
  - 5.3. Generar un nombre per simular la bola amb la funció `GeneraNombre`.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció `ImprimirBolesSortides` per imprimir els nombres de les bales que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció `Escrutini` per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment `ImprimirComentari`.
7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa...".
8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0.5 punts)**

Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `Nom_Jugador` (cadena de 50 caràcters).  
Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeixi l'identificador del cartó i el nom del jugador separant-los amb el caràcter `'.'`.

**Exercici 9 (1 punt)**

Escriviu la funció `Elevar` que retorni el resultat d'elevat un nombre al quadrat. La funció tindrà 2 paràmetres del tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realitzar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.  
NOTA: El valor de la base ha d'estar inclos en l'interval [-46.340,46.340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada `Es_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el numero de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

int copie (not), note [diputat], ~~date~~ ident, execution = 0  
copie  
note de cda  
note  
diputat  
note (1)

Wähle Erstwähler Votenzahl dem resultate[2])

int i;

do {  
cout <  
improt  
ig

for (i=0; i < dim; i++)  
resultats[N-2] += i;

{ while (

int I\_min ( int UL, int den ) {

int i; ( min, sum = 0;

min = N-2;

for (i=0; i < dim; i++)  
if (N-2 < min)

?

for (i=0; i < dim; i++) {

# Onaments d'Informàtica (103806)

Curs 2020-2021

Examen Parcial (23 de Novembre de 2020)

Grup:

Nom estudiant: BOUVE THOMAS JONAS NIU: 65 28 ASO

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherència entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartolls tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartell. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadaescú.

Exercici 1 (1 punt)

Fer un procediment anomenat CrearCartell per a generar un cartell. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1'1 a 15,
- 5 nombres del 16 al 30,
- 5 nombres del 31 al 45,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartell) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- int GenerarNombre(int min, int max, int a[], int dim)  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- void Ordena (int a[], int dim)  
Ordena l'array a de dimensió dim de menor a major.

$$17 + 17 = \underline{\underline{34}}$$
$$5 + 5 = \underline{\underline{10}}$$

Exercici 2 (1 punt)

Fer la funció anomenada ArraysIguals per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà True en el cas que siguin iguals i False en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$113 \div 50 = 2$$

RESULTAT: 20

Exercici 3 (1 punt)

Fer un procediment anomenat ImprimirBolesSortides per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contingut informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

NOTA: Recordeu que els números de les boles van de 1 a 75.

#### Exercici 4 (1 punt)

Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contindrà informació dels nombres que han sortit. En l'array bombó si el valor d'una determinada posició (index) és un 1, vindrà dir que la bola correspondent ha sortit i si el és un 0, vindrà dir que aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

#### Exercici 6 (2.5 punts)

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

```
Suposeu que tenui els següents procediments i funcions ja implementats:  
void MenuPrincipal()  
{  
    cout << " --- Menu Principal --- " << endl;  
    cout << "1.- Asignar cartrons " << endl;  
    cout << "2.- Jugar" << endl;  
    cout << "3.- Marcador" << endl;  
    cout << "4.- Sortir" << endl;  
}  
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Assignar cartons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)  
Inicialitza el vector a de dimensió dim al valor v
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main() ) que segueixi els següents passos:

1. Declarar (o corresponguï) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (o corresponguï) els arrays necessaris per representar els cartons de dos jugadors anomenats cartroPlayer1 i cartroPlayer2. Declarar també un array, anomenat bombo, per controlar quins noms han sortit i quins no.
3. Utilitzar el procediment MenuPrincipal() i llegir l'opcio escollida.
4. Si l'opcio és 1, generar dos cartons (un per a cada jugador) seguint els passos següents:
  - 4.1 Inicialitzar els cartons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2 Utilitzar el procediment CreateCartro per crear els cartons del jugador 1 i del jugador 2.
  - 4.3 Comprovar si els dos cartons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opcio és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartons (s'ha entrat a l'opcio 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els noms de les bales que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
6. Si l'opcio és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
7. Si l'opcio és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa..."
8. Qualsevol altra opcio, escriure el missatge: "Opcio no permesa".
9. Repetir els passos 3 a 9, fins que l'opcio del menú escollida sigui la 4.



**Exercici 7 (0,5 punts)**

Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `Nom_Jugador` (cadena de 50 caràcters). Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeixi l'identificador del cartó i el nom del jugador separant-los amb el caràcter `'.'`.

**Exercici 9 (1 punt)**

Escrive la funció `Elevar` que retorna el resultat de elevar un nombre al quadrat. La funció tindrà 2 paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'intervall [-46.340,46.340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada `Es_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

# Onaments d'Informàtica (103806)

Curs 2020-2021

Examen Parcial (23 de Novembre de 2020)

Grup:

Nom estudiant:

Heather Goldstein NIU: 2987229

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartells tenen un total de 25 nombres. En aquest tipus de bingo, es van treient boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartell. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes parties com vulguin i caldrà portar el compte de quantes parties ha guanyat cadascú.

Exercici 1 (1 punt)

Fer un procediment anomenat CrearCartell per a generar un cartell. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 11 al 15,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.
- 5 nombres del 16 al 30,
- 5 nombres del 31 al 45,

El procediment rebrà un array (que serà el cartell) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- int GenerarNombre (int min, int max, int a[], int dim)  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- void Ordena (int a[], int dim)  
Ordena l'array a de dimensió dim de menor a major.

$$200 + 60 = 260$$
$$100 \times 10 = \underline{\underline{1000}}$$

Exercici 2 (1 punt)  
Fer la funció anomenada ArraysIguals per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà True en el cas que siguin iguals i False en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$30 \times 40 = 1200$$

Resultat: 1200

Exercici 3 (1 punt)

Fer un procediment anomenat ImprimirBolesSortides per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contindrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

NOTA. Recordeu que els números de les boles van de 1 a 75.

#### Exercici 4 (1 punt)

Fer una funció anomenada `Escrutini` per saber si un cartó ha aconseguit Bingo. La funció rebrà un array que serà el cartó, la dimensió del cartó i un array (bombo) que contingrà informació dels nombres que han sortit. En l'array bombo si el valor d'una determinada posició (`index`) és un 1, voldrà dir que la bola correspondent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

La funció retornarà `true` en el cas que s'hagi aconseguit Bingo i `false` en cas contrari.

`}`

#### Exercici 6 (2,5 punts)

Suposeu que teniu els següents procediments i funcions ja implementats:

```
i
void MenuPrincipal()
{
    cout << " --- Menu Principal --- " << endl;
    cout << "1.- Asignar cartones " << endl;
    cout << "2.- Jugar" << endl;
    cout << "3.- Marcador" << endl;
    cout << "4.- Sortir" << endl;
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Asignar cartons, 2. Jugar, etc.

voind InicialitzarArray(int a[], int dim, int v)

Inicializa el vector a de dimensió dim al valor v

```
int GenerarNombre(int min, int max, int a[], int dim)
Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim
```

Fer un programa complet (declaracions globals i funció main()) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartons de dos jugadors anomenats `cartroplayer1` i `cartroplayer2`. Declarar també un array, anomenat `bombo`, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment `MenuPrincipal()` i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartons dels jugadors tot a 0 amb el procediment `InicialitzarArray`.
  - 4.2. Utilitzar el procediment `CreateCarto` per crear els cartons del jugador 1 i del jugador 2.
  - 4.3. Comprovar si els dos cartons són diferents utilitzant la funció `ArraysIguals`. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array `bombo` a 0 amb el procediment `InicialitzarArray`.
  - 5.3. Generar un nombre per simular la bola amb la funció `GenerarNombre`.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció `ImprimirBolesSortides` per imprimir els nombres de les bales que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció `Escrutini` per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment `ImprimirComentari`.
7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa...".
8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0,5 punts)**  
Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i nom `Jugador` (cadena de 50 caràcters).

Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeixi l'identificador del cartró i el nom del jugador separant-los amb el caràcter `.`.

**Exercici 9 (1 punt)**  
Escriure la funció `Elevar` que retorna el resultat d'elevar un nombre al quadrad. La funció tindrà dos paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornara un 1 si s'ha pogut realizar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'intervall [-46.340, 46.340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada `Es_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

cut Index

1500824

```
{  
    int posmin = 0;  
    int min = v[0];  
    for(int i=0; i < DIM; i++)  
    {  
        if(array[i] == 0) if((array[i]) < min)  
    }
```

1500824

$\{ \text{for } (\text{if } i = 0 \text{ then } \text{sum} + 1)$

$\{ \text{for } (\text{if } i > 0 \text{ then } \text{sum} + i)$

---

0:00

1532564

case 3: Escutini(2,20) & Escutini "cont cc" si no has set  
Escutini no sets a "mano" "cc endl;" ~~if (escutini != tots[0][2])~~  
~~cont cc, Escutini no esta set" "cc endl;~~

default: cont cc" Opcion no permitida" "cc endl;  
break;

} While (opcion != 4)

1532564

El void magical

int Dim1  
int Space1  
int Dim2  
int Space2

int Pentaheptad(Vo)

int min1, max1

int min2, max2

int min3, max3

void CntrOpC10

void SWI1ch(C0PC10)

void MnuPrc1eP1C1()

int CntrOpC10(Vo)

int min1, max1

int min2, max2

int min3, max3

int min4, max4

int min5, max5

int min6, max6

int min7, max7

int min8, max8

int min9, max9

int min10, max10

int min11, max11

int min12, max12

int min13, max13

int min14, max14

int min15, max15

int min16, max16

int min17, max17

int min18, max18

int min19, max19

int min20, max20

int min21, max21

int min22, max22

int min23, max23

int min24, max24

int min25, max25

int min26, max26

int min27, max27

int min28, max28

int min29, max29

int min30, max30

int min31, max31

int min32, max32

int min33, max33

int min34, max34

int min35, max35

int min36, max36

int min37, max37

int min38, max38

int min39, max39

int min40, max40

int min41, max41

int min42, max42

int min43, max43

int min44, max44

int min45, max45

int min46, max46

int min47, max47

int min48, max48

int min49, max49

int min50, max50

int min51, max51

int min52, max52

int min53, max53

int min54, max54

int min55, max55

int min56, max56

int min57, max57

int min58, max58

int min59, max59

int min60, max60

int min61, max61

int min62, max62

int min63, max63

int min64, max64

int min65, max65

int min66, max66

int min67, max67

int min68, max68

int min69, max69

int min70, max70

int min71, max71

int min72, max72

int min73, max73

int min74, max74

int min75, max75

int min76, max76

int min77, max77

int min78, max78

int min79, max79

int min80, max80

int min81, max81

int min82, max82

int min83, max83

int min84, max84

int min85, max85

int min86, max86

int min87, max87

int min88, max88

int min89, max89

int min90, max90

int min91, max91

int min92, max92

int min93, max93

int min94, max94

int min95, max95

int min96, max96

int min97, max97

int min98, max98

int min99, max99

int min100, max100

int min101, max101

int min102, max102

int min103, max103

int min104, max104

int min105, max105

int min106, max106

int min107, max107

int min108, max108

int min109, max109

int min110, max110

int min111, max111

int min112, max112

int min113, max113

int min114, max114

int min115, max115

int min116, max116

int min117, max117

int min118, max118

int min119, max119

int min120, max120

int min121, max121

int min122, max122

int min123, max123

int min124, max124

int min125, max125

int min126, max126

int min127, max127

int min128, max128

int min129, max129

int min130, max130

int min131, max131

int min132, max132

int min133, max133

int min134, max134

int min135, max135

int min136, max136

int min137, max137

int min138, max138

int min139, max139

int min140, max140

int min141, max141

int min142, max142

int min143, max143

int min144, max144

int min145, max145

int min146, max146

int min147, max147

int min148, max148

int min149, max149

int min150, max150

int min151, max151

int min152, max152

int min153, max153

int min154, max154

int min155, max155

int min156, max156

int min157, max157

int min158, max158

int min159, max159

int min160, max160

int min161, max161

int min162, max162

int min163, max163

int min164, max164

int min165, max165

int min166, max166

int min167, max167

int min168, max168

int min169, max169

int min170, max170

int min171, max171

int min172, max172

int min173, max173

int min174, max174

int min175, max175

int min176, max176

int min177, max177

int min178, max178

int min179, max179

int min180, max180

int min181, max181

int min182, max182

int min183, max183

int min184, max184

int min185, max185

int min186, max186

int min187, max187

int min188, max188

int min189, max189

int min190, max190

int min191, max191

int min192, max192

int min193, max193

int min194, max194

int min195, max195

int min196, max196

int min197, max197

int min198, max198

int min199, max199

int min200, max200

int min201, max201

int min202, max202

int min203, max203

int min204, max204

int min205, max205

int min206, max206

int min207, max207

int min208, max208

int min209, max209

int min210, max210

int min211, max211

int min212, max212

int min213, max213

int min214, max214

int min215, max215

int min216, max216

int min217, max217

int min218, max218

int min219, max219

int min220, max220

int min221, max221

int min222, max222

int min223, max223

int min224, max224

int min225, max225

int min226, max226

int min227, max227

int min228, max228

int min229, max229

int min230, max230

int min231, max231

int min232, max232

int min233, max233

int min234, max234

int min235, max235

int min236, max236

int min237, max237

int min238, max238

int min239, max239

int min240, max240

# anaments d'Informatàtica (103806)

Curs 2020-2021

Examen Parcial (23 de Novembre de 2020)

Grup:

NIU: 6243730

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherentia entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bombo. Els cartrons tenen un total de 25 números. En aquest tipus de bingo, es van treure boles del bombo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartó` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del l'1 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 31 al 45,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- `int GenerarNombre (int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordene l'array a de dimensió dim de menor a major.

$$16 + 15 = 31$$

$$30 - 10 = \underline{20}$$

$$\begin{array}{r} 100 \\ \times 20 \\ \hline 200 \end{array} = 1400$$

Resultat : 1400

Exercici 3 (1 punt)

Fer un procediment anomenat `ImprimirBollesSortides` per imprimir els nombres de les boles que ja han sortit. El procediment rebrà un array i la seva dimensió. L'array contindrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (index) és un 1, voldirà dir que la bola correspondent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

NOTA: Recordeu que els números de les boles van de 1 a 75.

L'exercici 4 (1 punt)  
Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retornarà `true` en el cas que siguin iguals i `false` en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

**Exercici 4 (1 punt)**  
Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que continguda informació dels nombres que han sortit. En l'array bombo si el valor d'una determinada posició (índex) és un 1, voldrà dir que la bola correspondent ha sortit i si el és un 0, voldrà dir que a aquella bola no ha sortit.

La funció retornarà true en el cas que s'hagi aconseguit Bingo i false en cas contrari.

**Exercici 6 (2,5 punts)**  
Suposeu que teniu els següents procediments i funcions ja implementats:

```
void Menuprincipal()
{
    cout << "---- Menu Principal ----" << endl;
    cout << "1. - Asignar cartrons" << endl;
    cout << "2. - Jugar" << endl;
    cout << "3. - Marcador" << endl;
    cout << "4. - Sortir" << endl;
}
```

Mostra per pantalla un menú amb les options del codi: 1. Assignar cartrons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)
```

Inicialitza el vector a de dimensió dim al valor v

```
int GenerarNombre(int min, int max, int a[], int dim)
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main()) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartrons de dos jugadors anomenats cartróPlayer1 i cartróPlayer2. Declarar també un array, anomenat bombo, per controlar quins noms han sortit i quins no.
3. Utilitzar el procediment Menuprincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
  - 4.1 Inicialitzar els cartrons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2 Utilitzar el procediment CrearCartró per crear els cartrons del jugador 1 i del jugador 2.
  - 4.3 Comprovar si els dos cartrons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. S'haurà de comprovar que s'han generat els cartrons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els noms de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
  6. Si l'opció és 3: imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
  7. Si l'opció és 4, imprimir el marcador final i mostrar el missatge: "Sortint del programa . . .".
  8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
  9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0.5 punts)**

Declarar un nou tipus de dades, TCartro, com un registre amb els camps: Ident\_Cartro (valor enter), Numeros (array de 25 enters) i Nom\_Jugador (cadena de 50 caràcters). Fer un procediment anomenat Imprimir que rebi com a paràmetre un registre del tipus TCartro i imprimeix l'identificador del cartró i el nom del jugador separant-los amb el caràcter '-'.

**Exercici 9 (1 punt)**

Escriviu la funció Elevar que retorni el resultat d'elevat un nombre al quadrat. La funció tindrà paràmetres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realitzar. La funció retornarà un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar inclos en l'intervall [-46.340,46.340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada Es\_MatPositiva que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

# Exercicis d'informàtica (1U38U6)

Curs 2020-2021

Examen Parcial (23 de Novembre de 2020)  
Nom estudiant: Vilma Josefa Nieves

Grup: NIU: 7216242

Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors idèntics. La funció rebrà dos arrays i la seva dimensió. Retorna `True` en el cas que siguin iguals i `False` en cas contrari.

NOTA: Podeu suposar que els dos arrays tenen la mateixa dimensió.

$$23 + 23 = 46$$
$$50 \times 2 = 100$$

Resultat: 100

Important: Recordeu que cal donar les millors solucions possibles en cada exercici. A més de funcionar correctament, el codi ha d'estar ben programat (codi clar, amb les instruccions més adients, sense operacions ni variables innecessàries, etc.)

Les funcions i procediments de les preguntes de la 1 a la 6 formen part d'un únic programa i, per tant, ha d'haver coherència entre les seves definicions, i la forma en què s'utilitzen en altres preguntes. El context del programa a desenvolupar, el teniu a continuació:

El Bingo de 75 boles és un joc d'atzar on hi ha 75 boles numerades de l'1 al 75 dins del bumbo.

Els cartons tenen un total de 25 números. En aquest tipus de bingo, es van treient boles del bumbo i es guanya un premi quan han sortit tots els nombres del nostre cartó. Això es coneix com a bingo o full House.

Volem fer un programa que permeti jugar al Bingo a dos jugadors. Els jugadors podran jugar tantes partides com vulguin i caldrà portar el compte de quantes partides ha guanyat cadascú.

Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartro` per a generar un cartó. S'han de generar 25 nombres del 1 al 75. S'ha d'assegurar que hi hagin:

- 5 nombres del 1 al 15,
- 5 nombres del 16 al 30,
- 5 nombres del 31 al 45,
- 5 nombres del 46 al 60 i
- 5 nombres del 61 al 75.

El procediment rebrà un array (que serà el cartó) i la dimensió de l'array. Retornarà l'array ple amb els números ordenats de menor a major.

UTILITZEU (no implementar) els següents procediments o funcions:

- `int GenerarNombre(int min, int max, int a[], int dim)`  
Genera un nombre aleatori entre min i max, i revisa que no estigui a l'array a de dimensió dim.
- `void Ordena (int a[], int dim)`  
Ordene l'array a de dimensió dim de menor a major.

$$600 + 40 = 640$$
$$300 - 5 = \boxed{295}$$

#### Exercici 4 (1 punt)

Fer una funció anomenada Escrutini per saber si un cartró ha aconseguit Bingo. La funció rebrà un array que serà el cartró, la dimensió del cartró i un array (bombo) que contindrà informació dels nombres que han sortit. En l'array bombo si el valor d'una determinada posició (index) és un 1, voldrà dir que la bola corresponent ha sortit i si el és un 0, voldrà dir que aquella bola no ha sortit.

La funció retornarà True en el cas que s'hagi aconseguit Bingo i False en cas contrari.

#### Exercici 6 (2,5 punts)

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenuPrincipal () {  
    cout << " --- Menu Principal --- " << endl;  
    cout << "1. - Asignar cartrons " << endl;  
    cout << "2. - Jugar" << endl;  
    cout << "3. - Marcador" << endl;  
    cout << "4. - Sortir" << endl;  
}
```

Mostra per pantalla un menú amb les opcions del codi: 1. Asignar cartrons, 2. Jugar, etc.

```
void InicialitzarArray(int a[], int dim, int v)  
{  
    for (int i = 0; i < dim; i++)  
        a[i] = v;  
}
```

Inicialitza el vector a de dimensió dim al valor v

```
int GenerarNombre (int min, int max, int a[], int dim)  
{  
    int nombre;  
    nombre = min + rand() % (max - min);  
    return nombre;  
}
```

Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim

Fer un programa complet (declaracions globals i funció main()) que segueixi els següents passos:

1. Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (on correspongui) els arrays necessaris per representar els cartrons de dos jugadors anomenats cartroPlayer1 i cartroPlayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment MenuPrincipal() i llegir l'opció escollida.
4. Si l'opció és 1, generar dos cartrons (un per a cada jugador) seguint els passos següents:
  - 4.1. Inicialitzar els cartons dels jugadors tot a 0 amb el procediment InicialitzarArray.
  - 4.2. Utilitzar el procediment CrearCartro per crear els cartrons del jugador 1 i del jugador 2.
- 4.3. Comprovar si els dos cartrons són diferents utilitzant la funció ArraysIguals. En cas que siguin iguals tornar a generar un dels dos.
5. Si l'opció és 2, implementa el joc seguint els passos següents:
  - 5.1. Shaurà de comprovar que s'han generat els cartons (s'ha entrat a l'opció 1). En cas contrari, donar un missatge d'error i tornar al menú principal sense fer res més.
  - 5.2. Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArray.
  - 5.3. Generar un nombre per simular la bola amb la funció GenerarNombre.
  - 5.4. Imprimir el nombre que ha sortit i utilitzar la funció ImprimirBolesSortides per imprimir els nombres de les boles que han sortit al llarg de la partida.
  - 5.5. Utilitzar la funció Escrutini per comprovar si algun dels dos jugadors ha aconseguit fer Bingo.
  - 5.6. En cas que no hi hagi cap guanyador tornar al punt 5.3
6. Si l'opció és 3: Imprimir el marcador de les partides que s'han jugat i mostrar el missatge generat pel procediment ImprimirComentari.
7. Si l'opció és 4: Imprimir el marcador final i mostrar el missatge: "Sortint del programa . . .".
8. Qualsevol altra opció, escriure el missatge: "Opció no permesa".
9. Repetir els passos 3 a 9, fins que l'opció del menú escollida sigui la 4.



**Exercici 7 (0,5 punts)**

Declarar un nou tipus de dades, `TCartro`, com un registre amb els camps: `Ident_Cartro` (valor enter), `Numeros` (array de 25 enters) i `Nom_Jugador` (cadena de 50 caràcters). Fer un procediment anomenat `Imprimir` que rebi com a paràmetre un registre del tipus `TCartro` i imprimeix l'identificador del cartó i el nom del jugador separant-los amb el caràcter ‘‘.

**Exercici 9 (1 punt)**

Escriviu la funció `Elevar` que retorni el resultat d'elever un nombre al quadrat. La funció tindrà parametres de tipus int. El primer serà la base i el segon serà el resultat de l'operació en el cas que es pugui realizar. La funció retornarà un 1 si s'ha pogut realizar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha de estar inclos en l'intervall [-46.340,46.340] per evitar overflow.

**Exercici 8 (1 punt)**

Fer una funció anomenada `Es_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número de columnes de la matriu, i retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

## #define DIPUTATS 135

```

int Vots [DIPUTATS], Resultat [DIPUTATS]; eleccio, parlamentari, roll, fuet
bool escrutini = false;
InicialitzarArray (Vots, DIPUTATS, 0);
do {
    MenuPrincipal ();
    cin >> eleccio;
    switch (eleccio) {
        case 1: while ((parlamentari > DIPUTATS) & & (parlamentari <= 0))
            cin >> parlamentari;
            Vots [parlamentari] = votar (0, DIPUTATS);
            break;
        Case 2: if (NoZeroArray (vots, DIPUTATS)) {
            InicialitzarArray (Resultat, DIPUTATS);
            Escrutini (vots, DIPUTATS, Resultat);
            escrutini = true;
            for (int i = 0; i < DIPUTATS; i++) cout << "Parlamentari " << i << vots[i];
            cout << endl;
        } else cout << "encara ...";
        Case 3: if (escrutini) {
            roll = IndexMinZeroArray (Resultats, DIPUTATS);
            fuet = IndexMaximArray (Resultat, DIPUTATS);
            cout << "la Fuet... " << fuet + 1 << "... " << roll + 1;
            else cout << "encara no hi ha ...";
            default: "OPC. no permeta"
        }
    }
}

```

```
int NormaVector (float x1, float y1, float x2, float y2, float &norma) {
    float res;
    res = sqrt ( pow (x2 - x1, 2) + pow (y2 - y1, 2));
    if (res == 0) return 0;
    else {
        return 1;
        norma = res;
    }
}
```

```
int Voter (int min, int max) {
    int p;
    do {
        cout << "Entre valor entre " << min << " e " << max
        cin >> p;
    } while (p > max || p < min);
}

} while (p < min || p > max);
```

Ex 6 int  $\sqrt{j}) \rightarrow 0$

1533089

for ( $j = a; j < m, it &$ ) {

if ((array[i] < array\_min) && (array[i] != 0))

{ pos = i; }

do while (array[i] == 0)

}  
i++;

~~for (i = 0, res = 0, it = 0;~~

~~int Vots[2]; int result[2];~~  
~~int option;~~

~~switch(option){~~

~~case 1: count = "Bürokrat" -~~

~~cin >> identifi;~~

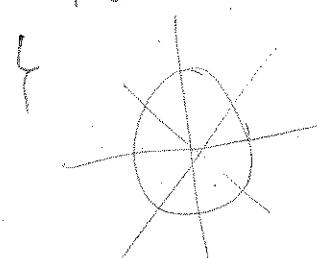
~~(do while (identifi <= 1 || identifi > PARLEMENTARIS) {~~

~~break~~

~~int i;~~

~~for (i = 1;~~

~~resultats[Vots[i] - 1]++;~~



Ex: 10

o  $\frac{sobri}{2} \geq pres tec \rightarrow S$

o  $\frac{sobri}{2} \geq pres tec \& \frac{sobri}{2} \leq pre$

o  $sobri \geq pres \& \frac{sobri}{2} \leq pre$

sofat res

~~res~~

res x = pow (X, 2)

res y = posw (

1. for loop  
 {  
     sum = 0;     for (i = 1; i <= n; i++) {  
         sum += a[i];  
     }  
     cout << sum;  
 }  
  
 2. for loop  
 {  
     sum = 0;     for (i = 1; i <= n; i++) {  
         sum += a[i];  
         cout << a[i];  
     }  
     cout << sum;  
 }  
  
 3. for loop  
 {  
     sum = 0;     for (i = 1; i <= n; i++) {  
         sum += a[i];  
         cout << a[i] << endl;  
     }  
     cout << sum;  
 }  
  
 4. for loop  
 {  
     sum = 0;     for (i = 1; i <= n; i++) {  
         sum += a[i];  
         cout << a[i] << endl;  
         cout << sum << endl;  
     }  
 }  
  
 5. for loop  
 {  
     sum = 0;     for (i = 1; i <= n; i++) {  
         sum += a[i];  
         cout << a[i] << endl;  
         cout << sum << endl;  
         cout << endl;  
     }  
 }  
  
 6. for loop  
 {  
     sum = 0;     for (i = 1; i <= n; i++) {  
         sum += a[i];  
         cout << a[i] << endl;  
         cout << endl;  
         cout << endl;  
     }  
 }  
  
 7. for loop  
 {  
     sum = 0;     for (i = 1; i <= n; i++) {  
         sum += a[i];  
         cout << a[i] << endl;  
         cout << endl;  
         cout << endl;  
         cout << endl;  
     }  
 }  
  
 8. for loop  
 {  
     sum = 0;     for (i = 1; i <= n; i++) {  
         sum += a[i];  
         cout << a[i] << endl;  
         cout << endl;  
         cout << endl;  
         cout << endl;  
         cout << endl;  
     }  
 }  
  
 9. for loop  
 {  
     sum = 0;     for (i = 1; i <= n; i++) {  
         sum += a[i];  
         cout << a[i] << endl;  
         cout << endl;  
         cout << endl;  
         cout << endl;  
         cout << endl;  
     }  
 }  
  
 10. for loop  
 {  
     sum = 0;     for (i = 1; i <= n; i++) {  
         sum += a[i];  
         cout << a[i] << endl;  
         cout << endl;  
         cout << endl;  
         cout << endl;  
         cout << endl;  
     }  
 }