

0186Fonaments d'Informàtica (103806) Curs 2020-

2021

Examen Parcial (23 de Novembre de 2020)

Grup:

Nom estudiant: JOSÉ ESTEBAN MARÍNEZ

NIU: 4177900

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 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 números 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 `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 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.

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

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 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

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;
```

```
}
```

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`; `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 següent 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 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 inclòs en l'intervall [-46,340,46,340] per evitar overflow.

Exercici 8 (1 punt)

Fer una funció anomenada `Es_MatPositiua` que rebi com a paràmetres: una matriu d'enters de 4 columnes, el número de files de la matriu, i el número 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.

```

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[1];

(o = 1; i < min & V[i] < o) {

 ++i; o = i; dim = i;

 min = V[0];

 ind = 0;

if (!min & ind)

 min = V[0];

res++

if (V[0] <= j & j <= V[i])

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

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

while (j < ~~dim~~ Outputs);

res = 0;
j++
Results[j] = res;

res++

if (V[0] <= j & j <= V[i])

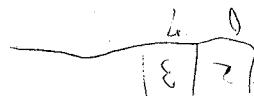
 for (i = 0; i < dim; i++)

} op

o = 0

res

j = 5



void ESCRUFN (int V[5], int dim, int Results[])

Exercicis d'informàtica (103806)

Curs 2020-2021

Examen Parcial (23 de Novembre de 2020)

Grup:

NIU: 5864644

Nom estudiant: Misti Paul Solomons

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 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 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 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 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.

$$15 + 15 = 30$$

$$35 - 10 = \underline{15}$$

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.

$$20 \times 4 = 80$$

$$30 + 20 = 50$$

Resultado: 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 (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; 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 `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 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 InicializarArray(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)
{
    return rand() % (max - min + 1) + min;
}
```

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 dels 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 `InicializarArray`.
 - 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 `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.

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 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.

J0186Fonaments d'informàtica (T0380)

Curs 2020-

2021

Examen Parcial (23 de Novembre de 2020)

Nom estudiant: AMBER VELITTEL

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. Retornar `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 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 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

- noms 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 cartró) 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 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.

$$85 \times 25 = 2125$$

$$100 - 35 = \boxed{65}$$

$$\begin{aligned} 72 + 72 &= 144 \\ \text{RESULTAT} &= 164 \end{aligned}$$

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 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 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 (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 `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".

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. Assignar cartons, 2. Jugar, etc.

```
void InicialitzarArray(int min, int max, int a[], int dim)
{
    int GenerarNombre (int min, int max, int a[], int dim)
    {
        // Implementació del generador de nombres aleatoris
    }
}

int GenerarNombre (int min, int max, int a[], int dim)
{
    // Implementació del generador de nombres aleatoris
}
```

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)
{
    int GenerarNombre (int min, int max, int a[], int dim)
    {
        // Implementació del generador de nombres aleatoris
    }
}

int GenerarNombre (int min, int max, int a[], int dim)
{
    // Implementació del generador de nombres aleatoris
}
```

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

- Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
- Declarar (en 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 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 `InicialitzarArray`.
 - 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:
 - 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.
 - Inicialitzar 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 nombres de les bolas 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, `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)

Escrure 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 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_MatPositiu` 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.

Ex. 2

```
int Votar (int min, int max)
```

```
{
    int vot;
    cout << "Entre valor minimo y maximo: ";
    cin >> vot;
    while ((vot < min) || (vot > max))
    {
        cout << "Error";
        cout << "Entre valor entre... ";
        cin >> vot;
    }
    return vot;
}
```

Ex. 4

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

```
{
    bool trabat = false;
    int i = 0;
    while ((i < dm) && (!trabat))
    {

```

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

```

```
            trabat = true;
        else

```

```
            i++;
        }
    }
}
```

```
if (!trabat)
```

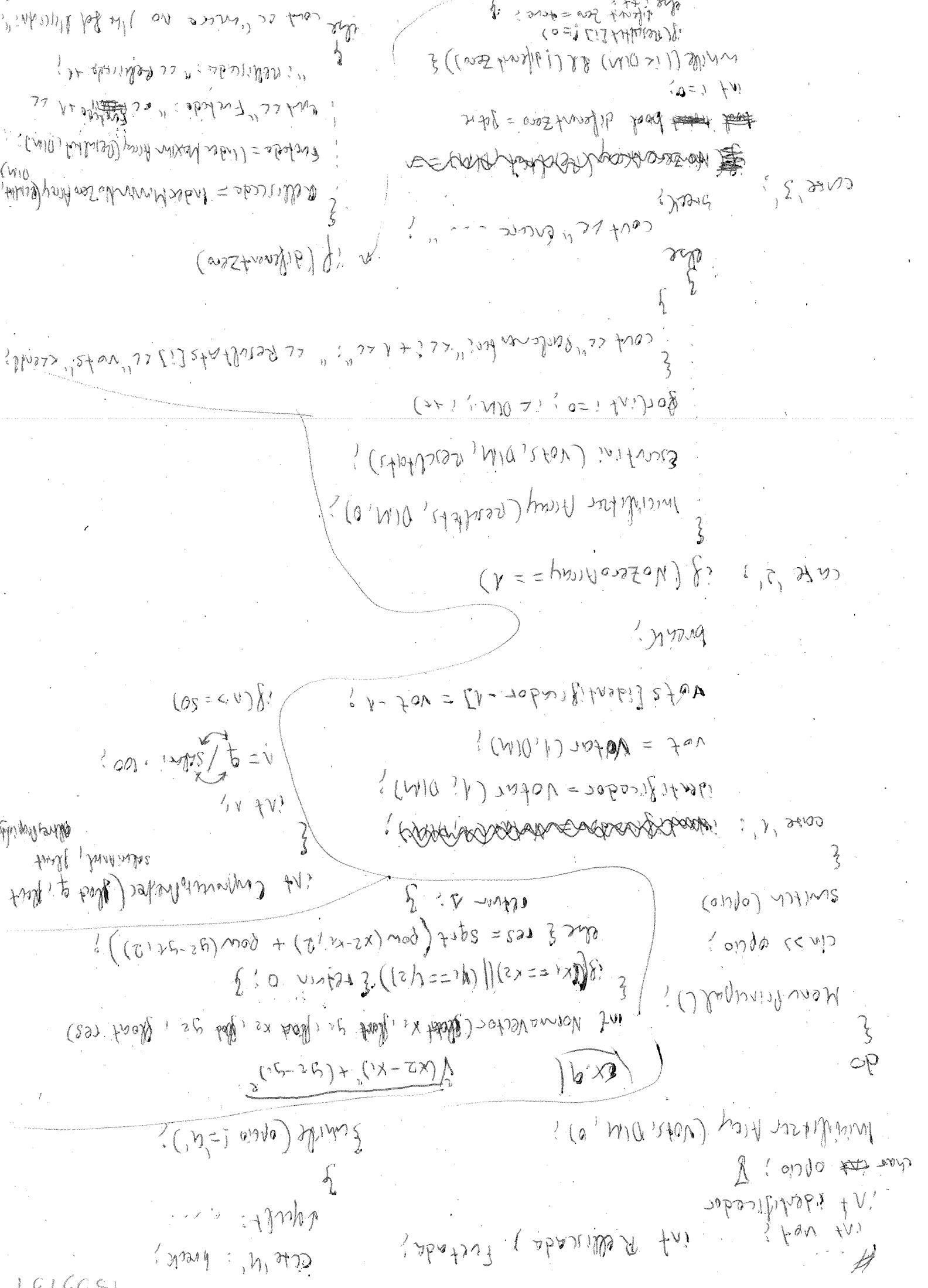
Ex. 5

```
void Escrutini (int votos[], int dm, int resultado[])
{
    for (int i = 0; i < dm; i++)
    {
        resultado[votos[i]]++;
    }
}
```

Ex. 6

```
int indexMinumNoZeroArray (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;
        }
    }
    return posmin;
}
```



Periodistes del Parlament

Fuetada \rightarrow millor { intervenció d'un parlamentari.
 Belluscada \rightarrow pitjor

Dipòtals \rightarrow 135

Ordenats alfabèticament \rightarrow assignant valors de l'1 a -135.

A ran

1 2

0

2

135

136

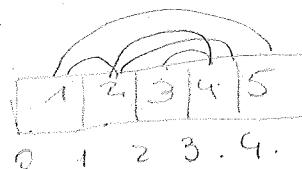
Todos deben votar

Alfabèticament ordenats. \rightarrow Alfabet 23.

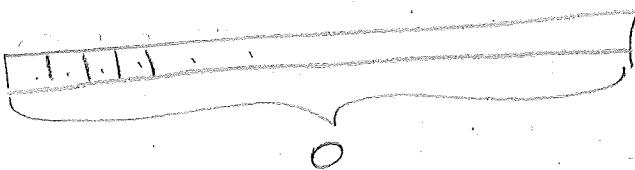
1
0

135

134



\rightarrow valor \rightarrow



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

1010101010101010 \rightarrow 0

$i=0$.

1010101010101010 \rightarrow 0



111111 \rightarrow 1.

else return 0;

else $\text{DustAcc} = \Delta_1$

if ($\text{Soda} \leq (\text{Preste} \times 10) / 100$) else

$\text{CO}_2 < \text{DustAcc}$

$\text{DustAcc} = P_1 + P_2$

else if ($\text{Soda} \leq (\text{Preste} \times 25) / 100$) || ($\text{Worder} / 2 \leq \text{Preste}$) {

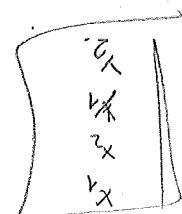
$\text{CO}_2 < \text{DustAcc}$

$\text{DustAcc} = P_1 + P_2$

} if ($\text{Soda} \leq (\text{Preste} \times 50) / 100$) || ($\text{Worder} / 2 \leq \text{Preste}$) {

$P_1 = 1.92 - 1.01 \times 0.25\% \rightarrow 50\%$
 $P_2 = 1.05 \rightarrow 50\%$
 $\text{DustAcc} = \text{Worder} + (\text{Preste} / 2)$

(x_1, y_1)
 (x_2, y_2)



0	1	2	3	4	5
1	2	3	4	5	

Onaments d'Informàtica (103806)

Curs 2020-2021
Examen Parcial (23 de Novembre de 2020)

Nom estudiant: James Davis Trudeau

Grup: NIU: 5089484

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 carttrons 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 carttró. 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 CrearCarttró per a generar un carttró. 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 carttró) 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.

$$4 \times 10 = \underline{40}$$
$$40 - 9 = \underline{31}$$

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.

$$145 - 20 = 95$$

$$\text{Resultat: } 100$$

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 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 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 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 5 (1 punt)

Exercici 5 (1 punt)

Fer un procediment anomenat ImprimirComentari per imprimir un comentari sobre el marcador. El procediment rebrà dos valors enteros 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".
- 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 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 cartons, 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:

- Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
- 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.
- 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:
 - Inicialitzar els cartons dels jugadors tot a 0 amb el procediment InicialitzarArray.
 - 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 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 nombres 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 permetuda".
 - 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 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 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.

Examen Parcial (23 de Novembre de 2020) Grup: NIU: 1887624
 Nom estudiant: FRANC琳 FLORES

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 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 cadaescú.

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.

- 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.

$$10 + 10 = 20$$

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

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.

$$30 + 40 = 70$$

$$12 \times 5 = 75$$

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 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.

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 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
    }
}
```

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)
```

```
voi d InicialitzarArray(int a[], int dim, int v)
```

```
int GenerarNombre(int min, int max, int a[], int 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 ArraysIquals. 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_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)

Escrure 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 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 retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

Menu →

```

100   int votos;
OPCIÓN 2 = do {
    {cout << "Introduce el voto << endl";
     cin >> Voto;
     {while ((Voto > min) && (voto < max)) →
      votar (int min, int max);
      for (int i = 0; i < dm; i++) {
          VOTOS
          {VOTOS[i];
      }
      }
      OPCIÓN 2 = int VOTOS[dm];
      NoCeroArray (dm);
      if (VOTOS < N_VOTOS)
          {Iniciarizar Array (dm);
           cout << VOTOS[i];
      }
      else
          {cout
      }
  }
  
```

```

int opción;
int votos [ ];
int voto;
{if (voto < min) || (voto > max)}}
  
```

```

{cout << "Introduce el voto << endl";
cin >> Voto;
{while ((Voto > min) && (voto < max)) →
do
  
```

white

int

salario;

IF (Salario <= 1000)

IF C salario - 100%

~~1000~~

10

```

OPCIÓN 3 =
do
{ IndiceMinimo NoCeroArray (Array [ ], dm);
  int IndiceMaximo Array [ ];
  }
  
```

cout << mensaje ~

while (escrutinio == True)

IF (escrutinio == false)

{cout << mensaje3

}

{

3 Wh

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

{

 if (v[i] > max) max = v[i];

 if (v[i] < min) min = v[i];

}

int min() {

 return min;

}

int max() {

 return max;

}

void print() {

 cout << "Min: " << min << endl;

 cout << "Max: " << max << endl;

}

int main() {

 int arr[5] = {1, 2, 3, 4, 5};

 int min, max;

 min = max = arr[0];

 for (int i = 1; i < 5; i++) {

 if (arr[i] < min) min = arr[i];

 if (arr[i] > max) max = arr[i];

 }

 cout << "Min: " << min << endl;

 cout << "Max: " << max << endl;

 return 0;

}

Onaments d'Informàtica (103806)

Curs 2020-2021

Examen Parcial (23 de Novembre de 2020)

Grup:

NIU: 8163427

Nom estudiant: RONALD BROCKLEHURST

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 bumbo. Els cartells tenen un total de 25 nombres. En aquests 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 guanyyat cadascú.

Exercici 1 (1 punt)

Fer un procediment anomenat `CrearCartro` 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 l'1 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 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)`
Ordene l'array a de dimensió `dim` de menor a major.

$$31 + 31 = 62$$

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

$$\begin{aligned} & 98 + 3 = 101 \\ \text{RESULTAT: } & 126 \end{aligned}$$

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 contingrà 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 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)
```

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 `CreaCartró` 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, `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 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 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.

1528461

~~100 → 37~~

~~Vote[i] = 37~~

~~Resultat [37] ++;~~

How Healthcare function

$a + n \cdot 0 = m$

$n = 0$

Onwards

($a + 0 = a$) to m

($a + 0 = a$) to

($a + 0 = a$) to

12881

The 3500 ft. level has been reached.

Wenckebach
D

卷之三

1. $\frac{d}{dt} \int_{\Omega} u^2 dx = -2 \int_{\Omega} u \cdot \nabla u dx$
2. $\frac{d}{dt} \int_{\Omega} |\nabla u|^2 dx = -2 \int_{\Omega} u \cdot \nabla |\nabla u| dx$

the dimensions of the vessel were
as follows: length 100 ft., width 20 ft., height 12 ft.

Yesterdays rain has caused a great deal of damage to the roads and fields. The water has washed away many trees and shrubs, and has caused flooding in several areas. The damage is estimated to be over \$100,000.

1. $\frac{1}{2} \times 10^3$ kg/m^3 $\times 10^3$ N/m^2 $\times 10^3$ $\text{m} = 5 \times 10^9 \text{ N}$

2
= $\lim_{n \rightarrow \infty} (1 + \frac{1}{n})^n$

Op

卷之三

Onaments d'Informàtica (103806)

Examen Parcial (23 de Novembre de 2020)

Curs 2020-2021
Grup:
NIU: 2A76813

Nom estudiant: Hèrcules Gremine Padró

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 l'1 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 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.

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

$$800 \div 20 = 40$$

Resultat: 2

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 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 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 noms 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 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)
{
    int GenerarNombre(int min, int max, int a[], int dim);

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

Retornar 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. 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 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 / (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 parametre un registre del tipus `TCartro` i imprimeixi l'identificador del cartró i el nom del jugador separant-los amb el caracter ','.

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 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 numero 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.

1528864

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

total = false

for (int i = 0; i < total;

total = true

switch (dia)

case 1:

; break;

typedef struct

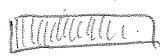
{

} class;

A B

IS29824

voltage drop



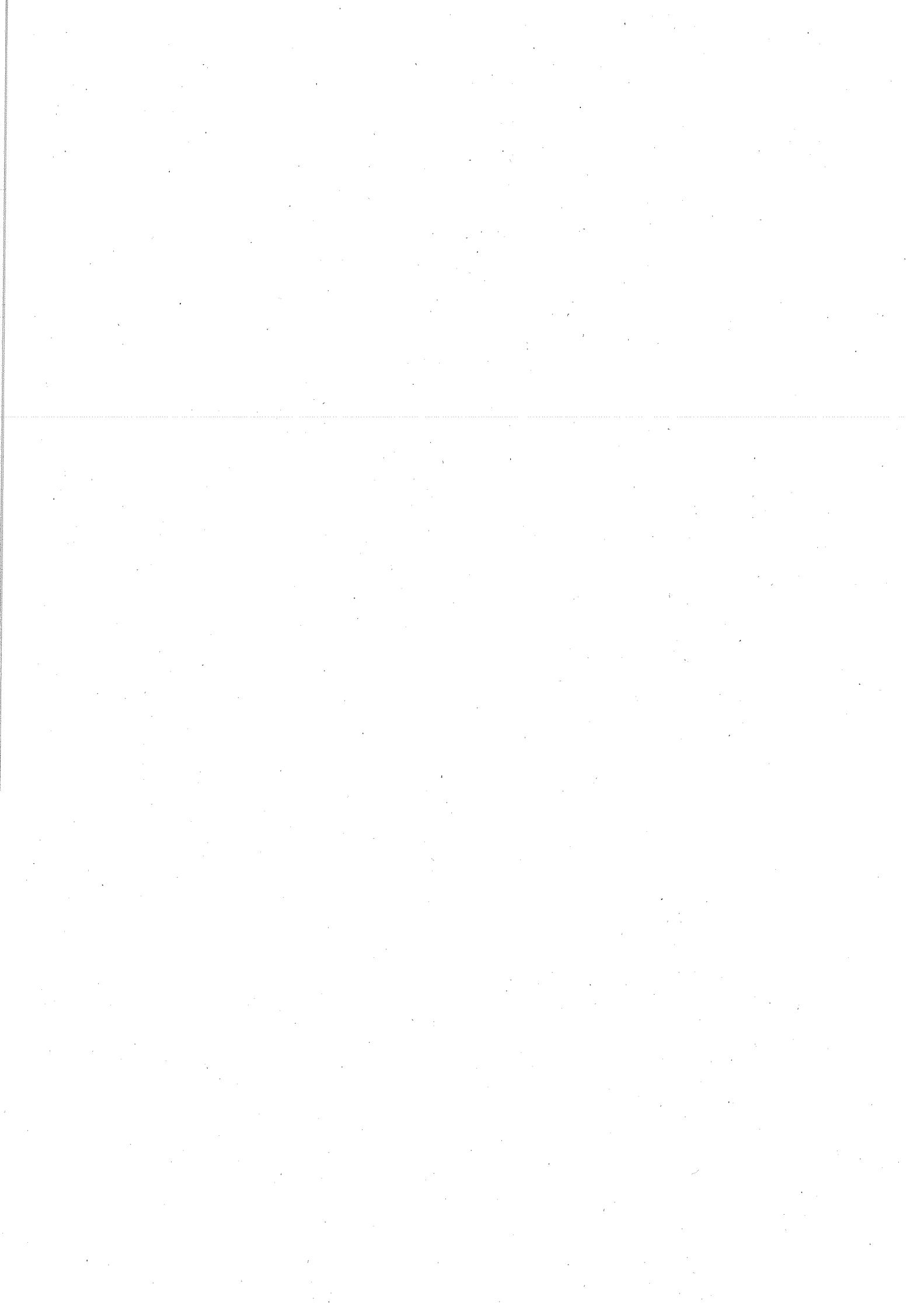
150

resistor:



12

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



anaments d'informàtica (i usos)

Examen Parcial (23 de Novembre de 2020)

inars 2020-2021
Grup:
NIU: 411 83 79

Nom estudiant: Nancy Stollings Geno

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 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 cartrons tenen un total de 25 nombres. En aquest tipus de bingo, es van tirant 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 caindrà portar el compte de quantes partides ha guanyat cadascú.

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 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 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 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.

$$5 + 14 = 23$$
$$23 \times 10 = \underline{120}$$

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.

Resultat: 54

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

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 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)`

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 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. 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 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'elevar 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 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 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)

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

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

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

exp min 10 1'is

150+

11's

50+ → 1's

11's

78+ → 12's

10+ → 11

$$x = \frac{-b}{a}$$

$$10\% \text{ d. } 10 \approx \frac{10 \cdot 10}{100} = 1$$

$$\frac{100-25}{100} = 75\%$$

Examen Parcial (23 de Novembre de 2020)

Nom estudiant: Charles Hill Dornat

Grup: 5269877
NIU: 5269877

Per la funció anomenada `IMPRIMIRBOLES` ha de ser 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 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 46 al 60 i
- 5 nombres del 61 al 75.
- 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,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.

$$11 + 7 = 18$$

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

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, 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 (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 << "---- 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 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 cartronPlayer1 i cartronPlayer2. 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 CrearCartron 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.

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 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 `Els_MatPositiuva` 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.

Fer la funció anomenada `ArraysIguals` per saber si dos arrays tenen tots els valors iguals. 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 tenuia 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 guanyyat 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 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)`
Ordene l'array a de dimensió dim de menor a major.

$$31 + 48 = 79$$

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

$$56 \times 1 = 56$$

$$20 + 56 = 76$$

Resultat: 76

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.

Exercici 4 (1 punt)
Fer una funció anomenada Escrutini per saber si el 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 de les bolas que han sortit. En l'array bombó 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 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. 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 ('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. 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 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'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 8 (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 '-'.

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

Exercici 9 (1 punt)

Escriviu 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 realitzar. La funció retornarà un 1 si s'ha pogut realizar l'operació i un 0 en cas contrari.

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 IR

def ISO

int Voter (int min, int max)

cout << "Enter voter entre " << min << " " << max;

cin >> a >> b;

while (a >= b)

file stream

int main()

switch (MemPr)

case A:

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 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 cartrons 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 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 `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 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.

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

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

Feu 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.

$$999 - 333 = 666$$

$$666 + 333 = 999$$

Resultat: 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 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 contingrà 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.

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;
}

```

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 es 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

Exercici 5 (25 punts)

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. 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 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 5 (25 punts)

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

```

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

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

int ArrayIguals (int a[], int b[])
{
    for (int i = 0; i < dim; i++)
        if (a[i] != b[i])
            return 0;
    return 1;
}

void CrearCarto (int a[])
{
    for (int i = 0; i < dim; i++)
        a[i] = GenerarNombre(1, 10);
}

```

Exercici 5 (25 punts)

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

```

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

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

int ArrayIguals (int a[], int b[])
{
    for (int i = 0; i < dim; i++)
        if (a[i] != b[i])
            return 0;
    return 1;
}

void CrearCarto (int a[])
{
    for (int i = 0; i < dim; i++)
        a[i] = GenerarNombre(1, 10);
}

int Escrutini (int a[], int dim, int bombos[])
{
    int bingo = 0;
    for (int i = 0; i < dim; i++)
        for (int j = 0; j < dim; j++)
            if (a[i][j] == 1)
                bingo++;
    if (bingo == dim)
        return 1;
    else
        return 0;
}

```

Exercici 5 (25 punts)

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

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 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.

10. int ComprovacioOposi(int anys, int mitjana)

and with

```
if (exp < exprmin(2^5/100)) { exp = exprmax(10/100); }  
parts = 1;
```

pants = t

else { if (lap < 50 / 100) gLoop

int. Recompte (int rotacions, int nro drets, int i)

rotacodes [numdeints]

esentini [nam Agents]

12

10.1
100

int. excretion 10

A small, dark, segmented insect larva, likely a caterpillar, shown from a side perspective.

Example (vectorial) numbers, such as

3) *Exodus (num 13:12, 13)*

Geometrical shapes

Tricentennial Area

~~1-500~~

Voiceover E:3 = Voter (high / max e)

(++? <numberts> : Q = ?) for

V6+actions

$$0 = 8 + 0.0 \cdot \left\{ \text{Temp}(\text{sum}) \right\} (0 = q + x_{\text{sum}})$$

$$g \in O(E) = \{g_{ij}\}_{ij} \in \mathbb{R}$$

Figure 1. A schematic diagram of the experimental setup.

2265251

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

void Recompete (int &votaciones, int dim votaciones,)

⑧ typedef struct {

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

int main ()

⑨ int EquacioPrimergrau (int a, int b, int &x,)
 if (a*x + b == 0) { return 1; }
 x = (-b)/(a);

else { if (

$$ax + b = 0$$

caro caro caro

3

larry larry

larry larry

(xow pur larry pur) noton pur

⑩

5

Recompte (int Votacions[], int Dim, int Escrivint)

```

}
int i;
for (i=0; i<Dim; i++)
}
```

4

```

    * EscriuInt (Votacions[i]) ++
}
```

7.

```

int main()
```

3

```

    * int Votacions[NCLIENTS]; int opcs; int zero;
    * int Recompte [NCOMERCIALS]; int rent; int i;
```

6.

```

int IndexMaximum (int codi[], int dim)
```

```

}
int i, valor1, valor2, max;
```

```

for (i=0; i<Dim; i++)
}
```

```

    * codi[i] = valor1 = codi[i];
    * valor2 = codi[i+1];

```

```

    if (valor1 >= valor2)
}
```

```

        switch (opco)
    }
```

```

        case 1:
    }
```

```

            ? $codi <> "Introduix el seu identificador: "
            ? $codi >> ident
    }
```

```

            ? while (ident < 0) || (ident > NCLIENTS)
            ? $codi >> ident
    }
```

```

            ? $codi >> ident
            ? Votacions[ident] = Voter (101, 112)
    }
```

```

}
}
```

1	2	3	4	5	6	7
3	6	9	3	2	4	7

3 void IncodificarArray (int cod[], int Dimm, int valor)

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

2

int Vtar (int min, int max)

```
{  
    int valor;  
    do {  
        cout << "Escribe un valor entre " << min << " y "  
            & max << endl;  
        cin >> valor;  
    } while ((valor >= min) && (valor <= max));  
    cout << "Vt. estás correctamente";  
    cout << endl;  
    else  
        cout << "Error: Valor fuera del intervalo";  
    cout << endl;  
    if (valor == min) || (valor > max)  
        return valor;  
}
```

4.

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

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 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 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.

$$239 - 100 = 139$$

$$239 + 100 = \underline{\underline{339}}$$

$$625 \times 2 = 625$$

$$625 + 152 = 777$$

Resultat: 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 contingrà informació sobre si un nombre ha sortit o no. Si el valor d'una determinada posició (`index`) és un 1, voltrà dir que la bola corresponent ha sortit i si el és un 0, voltrà 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 (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 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.

`void Escrutini(int** cartons, int dim, int* bombos, int index)`

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)`

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 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 `AreArraysEqual`. 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 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 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, `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 ‘:’.

Escriure la funció `Elevar` que retorna el resultat d'elevat 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 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.

1527046

$$ax+b=0$$

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

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

extra value of x

Examen Parcial (23 de Noviembre de 2020)

Grupo: NIU: 2A58560

Nombre estudiante:

Anne Smith Alvarado

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 GenerarNúmero (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.

$$50 \times 7 = 350$$

$$350 - 6^3 = \underline{\underline{287}}$$

Hacer la función llamada ArraysIguals 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.

$$5 \times 8 = 40$$

$$\text{Resultado} = 108$$

Ejercicio 3 (1 punto)

Hacer un procedimiento llamado ImprimirBolasSortides 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 `bombo` 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)

Suponga que tiene los siguientes procedimientos y funciones ya implementados:

```
void MenuPrincipal ()  
{ cout << "--- Menu 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.

```
void InicializarArray (int a [], int dim, int vr)  
Initializa el vector a de dimensión dim al valor vr
```

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

1. Declarar (donde corresponda) las constantes necesarias para que el programa sea fácilmente modificable.
2. Declarar (donde corresponda) los arrays necesarios para representar los cartones de dos jugadores llamados `cartonPlayer1` y `cartonPlayer2`. Declarar también un array, llamado `bombo`, para controlar qué números han salido y cuáles no.
3. Utilizar el procedimiento `MenuPrincipal ()` y leer la opción elegida.
4. Si la opción es 1, generar dos cartones (uno para cada jugador) siguiendo los pasos siguientes:
 - 4.1. Inicializar los cartones de los jugadores todo a 0 con el procedimiento `InicializarArray`.
 - 4.2. Utilizar el procedimiento `CrearCarton` para crear los cartones del jugador 1 y del jugador 2.
- 4.3. Comprobar si los dos cartones son diferentes utilizando la función `ArraysIguales`. En caso de que sean iguales volver a generar uno de los dos.
5. Si la opción es 2, implementa el juego siguiendo los siguientes pasos:
 - 5.1. 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.
 - 5.2. Inicializar el array `bombo` a 0 con el procedimiento `InicializarArray`.
 - 5.3. Generar un número para simular la bola con la función `GenerarNombre`.
 - 5.4. Imprimir el número que ha salido y utilizar la función `ImprimirBolesSortides` para imprimir los números de las bolas que han salido a lo largo de la partida.
 - 5.5. Utilizar la función `Escrutinio` para comprobar si alguno de los dos jugadores ha conseguido hacer Bingo.
 - 5.6. En caso de que no haya ningún ganador volver al punto 5.3
6. Si la opción es 3: imprimirl el marcador de las partidas que se han jugado y mostrar el mensaje generado por el procedimiento `ImprimirComentario`.
7. Si la opción es 4, imprimir el marcador final y mostrar el mensaje: "Saliendo del programa...".
8. Cualquier otra opción, escribir el mensaje: "Opción no permitida".
9. 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, $\{[12] \rightarrow \{101, \dots, 112\}\}$

150 clients, $\{0, \dots, 149\}$

```

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

```

```

typedef struct {
    - int escrutinio[150];
} Escrutinio;

```

2) int Votar(int min, int max)

```

{
    int x;
    cout << "Introduce valor entre " << min << " e " << max;
    cin >> x;
    while (x < min || x > max) {
        cout << "Error: valor fuera de l'intervalo";
        cin >> x;
    }
    cout << "Vot enres correcte";
    return x;
}

```

(Full)
esborrany

1. void recursive (int dim, int dim, int result, string path) {

 if (dim == 0) {
 cout << "Final result: " << result << endl;
 return;
 }
 for (int i = 0; i < 3; i++) {
 recursive(dim - 1, dim, result + to_string(i), path + to_string(i));
 }
 }

2. int main() {
 int n;
 cout << "Enter n: ";
 cin >> n;
 recursive(n, n, "", "");
 }

3. void recursive (int dim, int dim, int result, string path) {
 if (dim == 0) {
 cout << "Final result: " << result << endl;
 return;
 }
 for (int i = 0; i < 3; i++) {
 recursive(dim - 1, dim, result + to_string(i), path + to_string(i));
 }
 }

4. #include <iostream>

5. void recursive (int dim, int dim, int result, string path) {
 if (dim == 0) {
 cout << "Final result: " << result << endl;
 return;
 }
 for (int i = 0; i < 3; i++) {
 recursive(dim - 1, dim, result + to_string(i), path + to_string(i));
 }
 }

6. int main() {
 int n;
 cout << "Enter n: ";
 cin >> n;
 recursive(n, n, "", "");
 }

1525287

```
if (z == 0) {
    g->IndexMaximAny(resultat.resultat, Agents(Total))
    cout << "El començat guanyador es el " << g->
    if (z == 1) {
        cout << "Encara no s'ha fet el recompte de vots";
    }
}
```

~~default:~~

case h:

```
return V;
break;
```

default:

```
cout << "Opció no permet";
cin >> W;
```

full esborrany

contar intro langs i nota;
cintlangs > nota) int points = 0;
if (langs >= 15)
points = points + 5;
else if (langs >= 12's)
points = points + 3

1529328
if (nota > 9)
points = points + 5;
else if (nota >= 7) & (nota < 9)
points = points + 3;

15

AG - 0.400%

25%

$$\frac{25}{100} = 12.5\text{ langs}$$

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

NOTA: Recordeu que els dos arrays tenen la mateixa dimensió.

La funció rebrà dos arrays i la seva dimensió. Retornarà `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 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 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 `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 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.

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

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

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 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.

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 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.

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)
```

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 (o correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.

2. Declarar (o 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.

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.

Declarar un nou tipus de dades, `TCartró`, com un registre amb els camps: `Ident` – Cadena de 50 caràcters (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 `TCartró` i imprimeixi l'identificador del cartró el nom del jugador separant-lo amb el caràcter ‘,’.

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

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 retornara un 1 si s'ha pogut realitzar l'operació i un 0 en cas contrari.

Exercici 8 (1 punt)

Fer una funció anomenada `Es_MatPositiuva` 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 (ANALOGICO)

130 clientes 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;

}

$$\frac{5}{q} = x \quad ||$$

int a = float (float a, float b)

```

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

```

return punts Asignals;

s = Valid.

else

if (any-exp < s)

else

punts Asignals = B;

if (any-exp > n)

int exp-min = K;

int punts Asignals = O;

int int any-exp = L;

{

at compound Ops (float any-exp , float mitjana)

(E10)

If $\text{Equation prime form}$ (float a , float b)
 $\alpha x + \beta = 0$
 $\alpha x = -\beta$
 $x = -\frac{\beta}{\alpha}$
 If $(\alpha = 0) \& (\beta \neq 0)$
 $x = \infty$
 If $(\alpha \neq 0) \& (\beta = 0)$
 $x = 0$
 If $(\alpha \neq 0) \& (\beta \neq 0)$
 $x = -\frac{\beta}{\alpha}$

recomple

```
{ int i; aux;
```

1496505

```
for (
```

```
{
```

Votaciones [i] = aux;

Escrutini [aux-1] = Escrutini [aux-1] + V[i]

```
}
```


Examen Parcial (23 de Novembre de 2020)

Nom estudiant: Estrella Washington

NIU: 2360393

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 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 `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 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 cartró) 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 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.

$$90 - 35 = 55$$

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

Per la funció anomenada `ArraysIguals` per saber si dues arrays tenen els mateixos valors en els mateixos indrets. 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.

$$63 - 12 = 51$$

$$51 \times 20 = 1020$$

$$\text{Resultat: } 1020$$

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.

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 (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 4 (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)
{
    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 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 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'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 `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 `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 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 evit overflow.

1526616

 $\alpha = 0$

$$\begin{cases} ax+by=0 \\ 0+0=0 \\ 0+0=0 \end{cases}$$

$X=1$

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

 $\textcircled{1}$

$a=1 \quad b=0$

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

$\star 3, \oplus 5, \otimes$
Triangular array
rep

1. Only entries
2. dimensions
3. value either

Existencia de un vector nulo llamado parámetro

4) Hilbertianity
1PT
 1. May entries
 2. Dimension
 Between 1 si hay o
 Llamo si no hay o

2PT

Completo

1. Viscasas \rightarrow voto de cada elemp
2. Dimension viscasas
3. Elección \rightarrow vots per a cada cencial

3PT
3.0

Green Majority

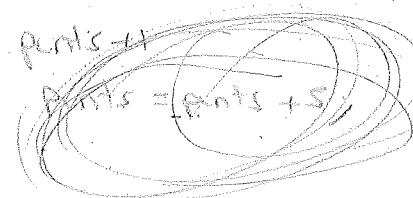
1. May entries
2. Dimension

Restar l'index de p's Maxim

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

1497995

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



if (a == 0)

mitjana = 10 años

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

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

$\sqrt{10 \text{ años}}$



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

$10 \text{ años} +$

$\text{any}s = 10;$

O's

$\text{any}s = 10$

$\text{any}s_1 = \text{any}s + \text{o}'s \cdot \text{om}$

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

~~if (mitjana > 9)~~

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

if

while ($ope1 == 4$)

def four: cout << "ope1 to perform"

case 4: break;

else { cout << "

case 1: if (four == 1) cout << "the set of recommonde" << "for 5";

case 2: if (four == 2) cout << "the set of recommonde" << "for 13";

case 3: if (four == 3) cout << "the set of recommonde" << "for 10";

case 4: if (four == 4) cout << "the set of recommonde" << "for 15";

Recommonde (Whactions, NumAGENTS, Accidents);
InitiateForPray (Escutin, NumAGENTS, 0);

if (vot == NumElements)

vot = HalfZeroPray (Whactions, NumElements);

case 2: break;

Whactions [vot] +=

vot = Voter (NumAGENTS MIN, NumAGENTS MAX);

else

Whactions = true;

case 1: if (NumElements) g8 (Whactions);

switch (ope1)

case 1: ope1;

def MenPunipole (A)

while (ICNUMCLIENT == 0)

while (ICNUMCLIENT == 0)

case 1: id;

int ope1; int vot;

bool flag = false;

Examen Parcial (23 de Novembre de 2020) Grup:
Nom estudiant: Miquel Rovira Heras NIU: 4343329

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 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 guanyyat 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.

$$7 \times 3 = 213$$

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

Per la funció anomenada `ArraysIguals` per saber si dos arrays tenen tous els valors idemius. 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.

$$90 - 8 = 82$$

$$\text{Resultat: } 125$$

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.

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 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. Assignar 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
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:

- 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 cartroplayer1 i cartroplayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit i 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:
 - Inicialitzar els cartons dels jugadors tot a 0 amb el procediment InicialitzarArray.
 - 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:
 - 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.
 - Inicialitzar 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 nombres de les botes 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.

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 `Elevant` 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.

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)

Nom estudiant: ~~Torrejón Pobres Beñat~~

NIU: ~~41083094~~

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 bumbo. Els cartolls 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 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.

$$4 \times 5 = 20$$
$$20 + 49 = \underline{69}$$

$$8 \times 8 = 64$$
$$64 - 18 = 46$$

$$\text{Resultat} : 46$$

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.

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 (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.

L'exercici (2,0 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 InicialitzarArry(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:

- Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
- 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.
- 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:
 - Inicialitzar els cartrons dels jugadors tot a 0 amb el procediment InicialitzarArry.
 - 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 ArxaysIguals. En cas que siguin iguals tornar a generar un dels dos.
 - Si l'opció és 2, implementa el joc seguint els passos següents:
 - Si l'opció és 2, 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.
 - Inicialitzar l'array bombo a 0 amb el procediment InicialitzarArry.
 - 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.

L'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-los amb el caràcter '-'.

Exercici 9 (1 punt)

Escriviu la funció Elevar² que retorni 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 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.

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 coerència 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 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 cadaescú.

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 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 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.

$$40 \times 20 = \underline{\underline{800}}$$

$$100 - 50 = \underline{\underline{50}}$$

Resultado : 95

$$12 \times 2 = 24$$

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, 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 (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 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. 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
```

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 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 retorni 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 `Fs_MatPositiva` que rebi com a paràmetres: una matriu d'enters de 4 columnes i 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 cadaescu.

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 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 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.

$$100 \times 20 = \underline{\underline{2000}}$$

$$90 + 15 = \underline{\underline{105}}$$

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.

$$12 + 10 = 114$$

$$100 - 10 = 90$$

Resultado : 90

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.

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, 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 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 cartoPlayer1 i cartoPlayer2. 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 cartons (un per a cada jugador) següint 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 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 cartró i el nom del jugador separant-los amb el caràcter `'`.

Exercici 9 (1 punt)
Escriure la funció `Elevar` que retorni el resultat de elevar 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 realitzar. 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 inclòs en l'interval [-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 retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

1497995

cin >> id;

bool troubt = false;
int i;

while (i < NCLIENT) { if (troubt)

{ if (id == 1000)
 else if (id == 2000)
 troubt = true;
 }
}

if (troubt)

if (vcis == 0)
 return 1;
else
 return 0;

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

$$\text{int } x; \\ 0x + b = 0;$$

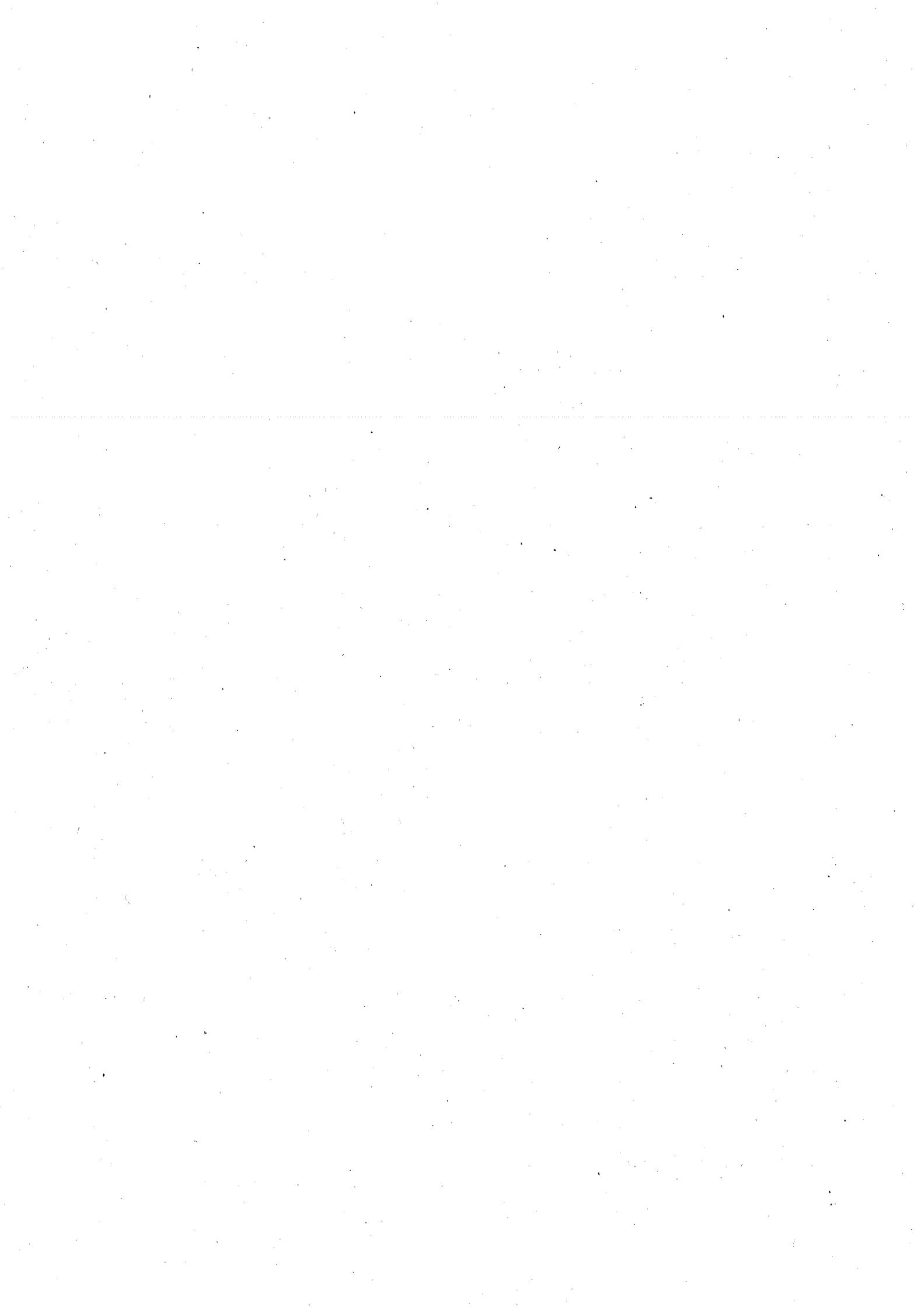
$$\text{if } (a == 0) \\ \text{return } \frac{x}{2};$$

$$\text{if } (b > 0), 3x = 0$$

$$x = -b/a, x = 0, x = 3$$

$$b = 0$$

$$1 = 0$$



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 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 l'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 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,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.

$$20 + 40 = \underline{60}$$

$$21 \times 15 = \underline{315}$$

Resultat : 315

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.

Per la funció anomenada `ArraysIguals` per saber si dos arrays tenen tous els valors iguals. 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 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.

Suposeu que teniu els següents procediments i funcions ja implementats:

```

Exercici 6 (25 punts)
{
    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)
{
    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
    }

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

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 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 bombó, per controlar quins nombres han sortit i quins no.
3. Utilitzar el procediment Menuprincipal () i llegir l'opció escollida.
4. Si l'opció es 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.

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 "A 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 7 (0,5 punts)

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 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 `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 retorna 1 si tots els valors de la matriu són positius o zero, i un 0 en cas contrari.

\$ \$ \$ \$ \$

do
| void newPrincipal();
| \Rightarrow opció;
| switch (opció);

case 1: do
| | case "Introduïx el seu identificador";
| | class newClient;
| | while (newClient <= 0 & newClient > clients)
for (int i = 0; i < clients; i++)
| | | voter Voter (NAME, MD5) (declarar-ho)!
| | | votacion[i] = voter[i];
| | }
| | break;

case 2: while (true & clients > 0) clients
res = MetodoZero votacion, votos = 0;
if (res == 1) votos = true;
cout << "Lucas no es pot fer caphe" << endl;

else
InitializarVotos (clients, ABENTS, 0);
Recuento (votacion, CLIENTS, res);
for (int i = 0; i < ABENTS; i++)

cout << "Reservat " << res << endl;
break;

case 3: metodo1, res;
votos votos = false;
while (true & clients > 0 & !votos)

res = MetodoUno (clients, ABENTS)

if (res == 1)

cout << "Lucas no es pot fer caphe" << endl;

else

$h = i \cdot \text{Winkel}$

hoch!

drehen! keine große, keine kleine Zelle

hoch!

Geht zu den kleinen Gruppen als die großen Gruppen + 10 Stück!

Um $\frac{1}{2} \pi$ umwenden (Etwas, Alles)

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 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 bumbo. Els cartons tenen un total de 25 nombres. En aquests 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 `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 l'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 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.

$$22 \times 53 = 423$$
$$423 - 7 = \underline{\underline{121}}$$

$$14 + 6 = 20$$
$$20 - 2 = 18$$

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 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 contendrà 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 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 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".
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.
5. Si l'opció és 2, implementa els passos següents:
 - 5.1. Si l'opció és 2, implementa els passos següents:
 - 5.1.1. 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 bombó 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 cartró i el nom del jugador separant-los amb el caràcter '-'.

Exercici 9 (1 punt)
Escriure la funció `Eleva2` que retorna el resultat d'elevant 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 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.

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;

 }

}

return ind;

{₁

{

{

!A+!

taut = true

autum Q)

} exp }

taut = true

autum A)

} (0 := L(7A)) p!

while ((taut || !taut) && (w > 1))

body taut = false

w := w - 1

} (w + !w) : taut = true

⑤

!w < w

cout << "Even: When you do the while"

while ((w < max) || (w < min))

⑥

Examen Parcial (23 de Novembre de 2020)Grup: 41
NIU: 5264370Nom estudiant: Jawne Falcone Locke

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 `CrearCartro` 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.

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.

$$4 \times 3 = 12$$

$$12 \times 2 = 36$$

Resolt: 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 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 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 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. Assignar cartrons, 2. Jugar, etc.

```
void InicializarArray(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 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 cartroplayer=1 i cartroplayer=2. 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 InicializarArray.
 - 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 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 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 imprimeix l'identificador del carro i el nom del jugador separant-los amb el caràcter ‘,’.

Exercici 9 (1 punt)

Escrive la funció `Elevar` que retorna el resultat d'elevat un nombre al quadrat. La funció tinc 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_MatPositiu` 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.

1492319

1. 9b columns

to define Newt-Column 9b

with

Type definition

Statement Newt-Column

declare Newt-Column

with 10 rows,

and 10 columns

{ 10 rows }

int RowIndex [Row Number]

for (int i = 0; i < NIA; i++)

and 10 columns, and rows, and 10 rows.

2. ~~for (int i = 0; i < NIA; i++) {~~

next (not the first one); } }

book cost = book;

int n;

while (cost > n) {

cost = cost - 1;

(cost -

else

if (n < min(WordCount))

cost = cost -

else

cost = cost - 1;

mark v;

for (

n = 1;)

if (v == 1)

6 molar

$$\frac{2}{9} = \frac{1}{x}$$

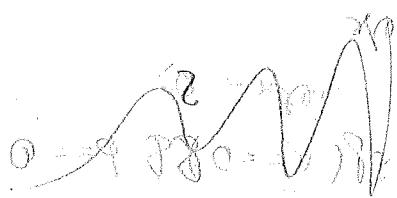
10 molar

28%

12 molar

$$(O=28) \frac{1}{8}$$

$$(C=22) \frac{2}{8}$$



10 molar < 12 molar

12 molar

28%

10 molar

$$(O=28) \frac{1}{8}$$

$\frac{1}{8} + \frac{1}{8} = \frac{1}{4}$

No. of solution 28

$$O=28 \text{ solution } 28 \leftarrow \frac{2}{9} - \frac{1}{x} \quad 28$$

```
#include <iostream>
```

1499509

```
#include ...
```

```
#define DIM 12
```

```
#define DMM 150
```

vs

```
using namespace std;
```

```
int main()
```

```
{ int vots; votsclients, votscomercials;  
int clients;  
int Votacions[DIM];  
int Escrutini[DIM];
```

```
void Votar (int min, int max) {
```

```
    int a;
```

```
    cin >> a;
```

```
do {
```

```
    if (a < min && a >= max) {
```

```
        cout << "Vot emes correctament";
```

```
    } else
```

```
        cout << "Valor fora de  
        l'interval";
```

```
} while (a <= min & & a >= max);
```

~~definir~~

~~if (array1[DIM] =~~

```
for (int i=0; i>0; i++)
```

~~if array1[i] :~~

```
    if (array1[i] = 0)
```

~~entza return 1;~~

~~else~~

~~return 0;~~

~~while~~

80



```
int IndexMaximArray (array2[DIM], array2[DIM])
```

8

Examen Parcial (23 de Novembre de 2020)

Nom estudiant:

Alice Kochiss Stewart

Grup: 43
NIU: 35509069

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 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 `CrearCartro` 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 l'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 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.

$$20 \times 10 = 200$$
$$200 + 50 = \underline{\underline{250}}$$

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 4 (1 punt)

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. 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 es 1, generar dos cartons dels jugadors tot a 0 amb el procediment `InicialitzarArray`.
- 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 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)

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 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.

$$ax + b = 0$$

$$\frac{-b}{x} = a$$

case 1:

$$c_{in} \gg iden;$$

~~while (iden > 849) {~~ ~~if (iden < 0)~~

$$c_{in} \gg iden;$$

$$ax = -b$$

$$-b/a = x$$

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

{

do ($\text{vot}[i] = \emptyset$)

($\text{vot}[i] \in \{1\}$)

$\text{vot}[i] \in \{2\}$

$\text{vot}[i] \in \{3\}$

3

2

$\text{vot}[i] \geq \text{esc}[i]$

}

int $f = 101;$

104

112

101

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

{

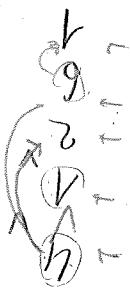
do

{

$\text{Escutini}[f]++;$

if ($\text{vot}[i] == f$)

8



return vector

out <<

$\text{while}(\text{min} \leq \text{max})$
 $\text{min} = \text{max}$

if ($\text{min} > \text{max}$) Error

do $\text{out} \ll \text{val}$, $\text{val} = \text{min}$, $\text{min} + 1$

End

* define N-CLIENTS 150

* define N-ACON 12

O - 149 To be refactored

Vofacto 150 clients

12 aquile commercial — will be commercial

+ older objective [101-112]

Hinchfield (with team)
define Client.h

A4013499

int main () {

int Voteriom [Clients] ; int oprio;

int Execution [Agents]; int identifier;

Initalization (Clients, Voteriom [0], 0)

MenuPrincipal()

(in >> oprio;

Switch (oprio) {

Case '1':

While (

Do

if (oprio != 4)

cin >> oprio;

Define Agents 12

#define Agents 159
int Temova_EAgents
int votobion [votb]

while ($\alpha < \min(1, \alpha + \text{min})$) {

 center = $\alpha \text{ mod } i$
 count = $i - \text{center}$

 min => a[i]
 }

for (i=0; i < Dim; i++)
 con Existence [Votabion [i] - minid] ++;

int i=0, a = array [a], min;
for (i=0; i < Dim; i++)
 if (Existence [i] > a)
 min = i;
 a = Existence [i];

Examen Parcial (23 de Novembre de 2020)**Nom estudiant:** Carolin Rose Kotzter**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 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 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 `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 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 cartró) 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 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.

$$20 \times 10 = 200$$

$$200 + 50 = \underline{\underline{250}}$$

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.

$$4 \times 3 = 10$$

$$10 \times 2 = 20$$

$$\text{Resolt: } 20$$

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, 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 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 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 cartoPlayer1 i cartoPlayer2. Declarar també un array, anomenat bombo, per controlar quins nombres han sortit 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.

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 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.

in array (array[], int num, int value)

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

{ array[i] = value / 10

value = value % 10;

for (i = 0; i >= 0; i--)

while (i < 0 && !break)

i++;
if (array[i] >= 0)
break; true;

void reverse (int char
vectors[], int num, int Escritor[])

{ int i;
for (i = 0; i < num; i++)
Escritor[vector[i]]++;
case 0 : Escritor[0]++;
case 1 : Escritor[1]++;

case 2 : Escritor[2]++;

VOTS
A
O
I
Q
P
J

CONCURSAN
S
S
9

1493112

2nd 1921 4W

operador A

operador 2

operador 3

$$\text{operador} A = x_2 - x_1$$

operador A = float(sqrt(float(operator A)))

$$\text{operador} 2 = x_2 - y_4$$

operador 2 = float(sqrt(float(operator 2)))

if (operator A == operator 2)

return 0;

else

}

D	P	E	G	T	K	J	E	I	B	O	A	0
0	1	0	1	1	1	0	1	1	1	0	1	1

Nom estudiant: Chad Matthews

Exercici

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 `CrearCartro` 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.

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.

$$20 \times 10 = 200$$

$$200 + 50 = \underline{\underline{250}}$$

$$4 \times 3 = 12$$

$$12 \times 2 = 38$$

Resultat: 38

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 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 6 (2.5 punts)

Suposeu que tenui 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;
}
```

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

```
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 segueix 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 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 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 8 (1 punt)
Escriure la funció `Elevar` que retorna el resultat d'elevat 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 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.

int IndexMinValue (int array[], int dim)

```
{  
    int i; minimum = array[0];  
    int pos;  
    for (i=0; i < dim; i++)
```

1497689

```
    if (array[i] < minimum)  
    {  
        minimum = array[i];  
        pos = i;  
    }  
}
```

Vals [Valor] 0 1 7 6 10 12 3 1 8 250

Repalaito [Cavamat] 1 2 4 1 6 2 2 ... 5 4 7 2 7 15

int Valor (int min, int max)

```
{  
    int valor;
```

(out >> "Entre Valor entre &: &: 44;

dim << 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 (at

if (array[i] == 0)

```
{  
    retorno = 0;
```

```
}
```

else

```
{  
    retorno = 1;
```

```
}
```

Popularity ($\text{Votes}[i] \rightarrow 1$) = $\text{Votes}[i]$

$$1 - \alpha$$

β

Popularity ($\text{Votes}[i] \rightarrow 1$) = $\text{Votes}[i]$

$$2 = r = \beta$$

$\beta = 1$

Popularity

1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9
0.52	1	2	3	4	5	6	7	8	9

Popularity

$\text{Popularity} (\text{Votes}[i], \text{Votes}[j], \dots, \text{Votes}[n])$

$\text{Popularity} = 0$

Word (Commonality ($\text{int Votes}, \text{word_words}$, int Popularity))

Popularity

$\text{Popularity} (\text{Votes}[i] \rightarrow 1) = \text{Votes}[i]$

Popularity

$$\alpha = 0$$

$$\beta = \text{Votes}[i]$$

Popularity ($\text{Votes}[i] \rightarrow 1$) = $\text{Votes}[i]$

$\alpha = 1$

$$\beta = 0$$

$(\alpha = \text{Votes}[i] = 0)$

Popularity ($\text{Votes}[i] \rightarrow 1$) = $\text{Votes}[i]$

$\alpha = 0$

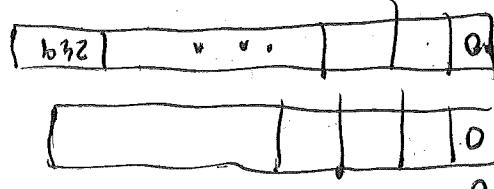
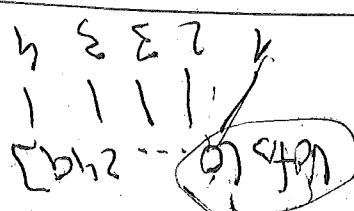
$$\alpha = 1$$

Value ($i > \text{dim}$)

int value = 1.

Popularity ($\text{Votes}[i] \rightarrow 1$)

No zero array (int array [int dim])



Popularity ($\text{Votes}[i]$)

Examen Parcial (23 de Novembre de 2020)

Nom estudiant: CASSIÉ SARDOUVAL

Curs 2020-2021

Grup:

NIU: 3565110

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 cadaescú.

Exercici 1 (1 punt)
Fer un procediment anomenat CrearCartró 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.

$$20 \times 20 = 400$$

$$7 \div 7 = \boxed{1}$$

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.

$$300 + 300 = 600$$
$$150 \times 2 = 300$$
$$\text{RESUM} = 100$$

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 contingrà 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.

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 6 (2.5 punts)

Suposau 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 cartons, 2. Jugar, etc.

```
void InicialitzarArray(int min, int max, 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 `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 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 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 9 (1 punt)

Escriure la funció `Elevar` que retorna el resultat d'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 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.

Vlaardingen School Niv: 1503148

char m[]; Escrutini: 12

int i; vot;

for (i=0; i<12; i++)

{
vot = Notaciones[i] - 101;
Escrutini[vot]++;
}

6.2.3 Escrutini + 101;

q. do while int i;

{
int main(void)
{
int i;
int votaciones[150];
int Escrutini[12];
char bole_recompte = false;
InitializarArray(bole_recompte);
InitializarArray(Escrutini, 12);
int i = IndexMaximArray(Escrutini, 12);
cout << "El comercial ganador es el de indice " << i << endl;
if (bole_recompte == true)
{
i = IndexMaximArray(Escrutini, 12);
cout << "El comercial ganador es el de indice " << i << endl;
}
else
cout << "Encara no se ha hecho recompte de votos." << endl;
break;
case 'S':
break;
default:
cout << "Opcion no permanece" << endl;
break;
}
while (m != 140);
return 0;

3
while (m != 140);
return 0;

8

8 & 8

$a = 0 \rightarrow \emptyset$
 $b = 0 \rightarrow \checkmark$
 $\sim b = 0 \rightarrow \infty$

Examen Parcial (23 de Novembre de 2020)

Nom estudiant: JOSE GARCIA

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 `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.

$$25 + 13 = \underline{38}$$
$$10 + 2 = \underline{12}$$

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. Retorna `true` en el cas que siguin iguals i `false` en cas contrari.

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

$$82 \times 4 = 336$$

$$\text{RESULTAT} = 12$$

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, 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 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 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)  
    {  
        //Retorna un enter aleatori entre min i max, controlant que no estigui a l'array a de dimensió dim  
        return rand() % (max - min + 1) + min;  
    }  
}
```

```
int InicialitzarArray(int a[], int dim, int 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 noms 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 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, `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 imprimeixi l'identificador del cartró i el nom del jugador separant -los amb el caracter '-'.

Exercici 9 (1 punt)

Escriure la funció `Elevar` que retorni el resultat d'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 realitzar. La funció retornarà un 1 si sha 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 espice int) notes [digitate], ~~dark~~ ident, execution = 0
Slope
slope notes
Notes
Digitate
Notes (N)
Notes (N)

nellimada
metoda

6698351

Wähle Erstellung (Vater[2] < dim, resultate[2]) {

int i;

do {
cout <<
current
i;

{ while () ;

int I Min (int UL, int dim) {

int i; i min, par = 0;

~~min~~ min = UL;

for (i=0; i<dim; i++) {
if (ig (N L I L K R S S min))
if (par = i);
}

Onaments d'Informàtica (103806)

Curs 2020-2021

Examen Parcial (23 de Novembre de 2020)

Grup:

Nom estudiant: BOUËT THOMAS TORÉS NIU: 6528150

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 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 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 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.

$$17 + 17 = \underline{\underline{34}}$$
$$5 + 5 = \underline{\underline{10}}$$

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.

$$45 + 45 = 90$$

$$113 - 50 = 2$$

RESULTAT: 70

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 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 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
```

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

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 es 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 6 (2.5 punts)

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 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: imprimint 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 retorni el resultat d'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 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 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.

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 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 bumbo. Els cartrons 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 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 `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 1'1 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 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)`
Ordene l'array a de dimensió dim de menor a major.

$$200 + 60 = \underline{260}$$
$$100 \times 10 = \underline{1000}$$

$$30 \times 40 = 1200$$

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; 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 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.

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 bombo 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 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.- 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 (o correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (o correspongui) 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 `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 `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 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 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 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 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.

1500824

cut Index _____

```
{  
    int posmin = 0;  
    int min = v[0];  
    for(int i=0; i < DIM; i++)  
    {  
        if(array[i] == 0) if((array[i]) < min)  
        {  
            posmin = i;  
            min = array[i];  
        }  
    }  
}
```

1500824

(++)(MID) $\int \phi = \int f(u) u^{\alpha} \{$
(++) (MID) $\int (u - 1) u^{\alpha} f(u) u^{\alpha} \{$

100

1532564

case 3: ~~Escutini~~ D120 & Escutini cont cc "Si no bus set
~~Escutini~~ no es el año " cont; ~~Si~~ ~~Escutini~~ no es el año
cont cc ~~Escutini~~ no es el set" cont;

default: cont cc "Opcion no permiso" cont;
break;

} While (opcion!=4)

1532564

an void magical

{ at Dim1
at DPCIO,
at PDBNADL
at PDBNADL(VOF);
at min, max;
int max[];

an PDBNADL(VOF);
at min, max;
int max[];

Int CiallTzar Array(VOFs[]);

void MenuPrint();

void CntrPcic();
do do {
switch (OPCIO)

SWITCH (OPCIO)

case 1: cout << "Indudeixi l'indentificador del paràmetre que voluteu!"

ch >> parametroLeft;

def Left(min, max);

VOFs(VOF);

break;

cas 2: void NoZeroArray (Dim1, VOFs[]);

if (WZeroArray == VOFs(VOF))

else Selection Escutin(Results[])

if (NoZeroArray(Dim1, Results[]))

else Selection Escutin(Results[])

break;

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 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 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 1'1 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 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.

$$16 + 15 = 31$$

$$30 - 10 = \underline{\underline{20}}$$

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.

$$100 + 60 = 160$$

$$\text{Resultat: } 1400$$

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 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 5 (2,5 punts)

Suposeu que teniu els següents procediments i funcions ja implementats:

```
void MenuPrincipal()
{
    cout << " --- Menu Principal --- " << endl;
    cout << " 1. - Assignar 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. 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
```

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 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 6 (2,5 punts)**
- Fer un programa complet (declaracions globals i funció main()) que segueixi els següents passos:
- Declarar (on correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
 - 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 noms han sortit i 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 InicialitzarArray.
 - Utilitzar el procediment CrearCarto per crear els cartons del jugador 1 i del jugador 2.
 - Comprovar si els dos cartons són diferents utilitzant la funció ArraysTquals. 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 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 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: "S'outint 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, `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 un nombre al quadrad. 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 realitzar l'operació i un 0 en cas contrari.

NOTA: El valor de la base ha d'estar incòs en l'interval [-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 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)
Nom estudiant: William Joseph Nieves

Grup: NIU: 7216242

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'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 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|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.

$$600 + 40 = 640$$

$$300 - 5 = \underline{295}$$

$$\begin{array}{r} 23 \\ \times 2 \\ \hline 46 \end{array}$$

$$\begin{array}{r} 50 \\ \times 2 \\ \hline 100 \end{array}$$

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 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 (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 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)
```

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 correspongui) les constants necessàries per tal que el programa sigui fàcilment modificable.
2. Declarar (o 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 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 CreaCartro 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 torna 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 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 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 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 de 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.

#define DIPUTATS 135

```

int Vots [DIPUTATS], Resultat [DIPUTATS], eleccio, parlamentari, rell, 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, 0);
            Escrutini (vots, DIPUTATS, Resultat);
            escrutini = true;
            for (int i = 0; i < DIPUTATS; i++) cout << "Parlamentari " << i + 1 << vots[i] <<
                " Vots " << endl;
            } else cout << "Encara ... ";
        Case 3: if (escrutini) {
            Rell = IndexMinimumZeroArray (Resultats, DIPUTATS);
            Fuet = IndexMaximumArray (Resultat, DIPUTATS);
            cout << "La Fuet... " << Fuet + 1 << "... " << Rell + 1;
            ?else cout << "Encara no hi ha ... ";
        default: "OPC. NO PERMESA";
    }
}
while (eleccio != 4);

```

```
int NormalVector (float x1, float y1, float x2, float y2, float &normal)
{
    float res;
    res = sqrt (pow (x2 - x1, 2) + pow (y2 - y1, 2));
    if (res == 0) return 0;
    else {
        return 1;
    }
    normal = res;
}
```

```
int Voter (int min, int max)
{
    cout << "Entre valeur entre " << min << " et " << max
        << endl;
    int p;
    do
    {
        cin >> p;
        while (p > max || p < min)
    }
```

```
}
```

Ex 6 int $\sqrt{j}) \rightarrow = 0$

1533089

for (j=a; !(arr[i], ite)) {

if ((array[i] < array-min) && (array[i] != 0))

{ pos = i; }

do while (array[i] == 0)

i++;

~~for (i=0, result, ite);~~

int Vots[2]; int Result[2];
int option;

Ex 7

switch(option){

case 1: count = "Parlement" -

"cin >> identif;"

(do while (count <= 1 || count > PARLEMENTARS) {

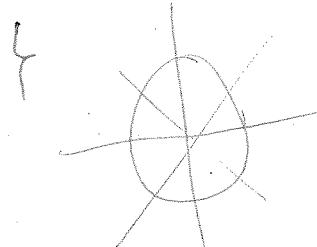
break

~~break~~

int i;

for (i; {

results[Vots[i]-1]++;



Ex 10

o $\frac{\text{salari}}{2} \geq \text{prestec} \rightarrow S$

o $\frac{\text{salari}}{3} \geq \text{prestec} \& \frac{\text{salari}}{2} < \text{pre}$

o $\text{salari} \geq \text{pres} \& \frac{\text{salari}}{3} < \text{pres}$

~~scott test~~

~~000000~~

res x = pow (X, 2)

res y = pow (

while (count < n) {
 cout << "Enter value" << endl;
 cin >> val;
 if (val < min || val > max) {
 cout << "Value is out of range" << endl;
 } else {
 cout << "Value is in range" << endl;
 sum += val;
 count++;
 }
 }
 cout << "Sum = " << sum << endl;
}

Ex. 2

while (count < n) {
 cout << "Enter value" << endl;
 cin >> val;
 if (val < min || val > max) {
 cout << "Value is out of range" << endl;
 } else {
 cout << "Value is in range" << endl;
 sum += val;
 count++;
 }
 }
 cout << "Sum = " << sum << endl;
}

Ex. 3

while (count < n) {
 cout << "Enter value" << endl;
 cin >> val;
 if (val < min || val > max) {
 cout << "Value is out of range" << endl;
 } else {
 cout << "Value is in range" << endl;
 sum += val;
 count++;
 }
 }
 cout << "Sum = " << sum << endl;
}