UTN FRBA - Algoritmos y Estructura de Datos – Comisión Ing. Pablo Mendez

Curso: Apellido y Nombres: Legajo: Fecha:

1)

```
void func(int v[], int tam)
{
    *v = 10;
    return;
}

int main()
{
    int *p = NULL;
    int x = 30;
    p = new int[3];
    p[1] = x;
    func(p,3);
    p[2] = *(p+1) + x;
    for (int i=0;i<3; i++)
        cout << *(p+i) << "  ";
    delete [] p;
    return 0;
}
```

La salida por pantalla es:

a) 10
b) 10 30 0
c) 10 30 null
d) 10 30 60
e) 30 31 null
f) 30 25 55
g) 30 25 56
h) Una dirección de memoria
i) Ninguna, no compila. JUSTIFICAR.

2)

```
struct Nodo
{
        int info; Nodo *sgte;
};

int main()
{
    Nodo *p= NULL; Nodo *aux;
    p = new Nodo();
    p->info = 1; p->sgte= new Nodo();
    p->sgte->info = 2; p->sgte->sgte = new  Nodo();
    p->sgte->sgte->info = 3; p->sgte->sgte->sgte= NULL;
    aux= p;
    p = p->sgte;
    p->sgte->sgte = aux;
    aux->sgte = NULL;
    aux = p;
    while (aux)
    {
        cout << aux->info << "; ";
        aux= aux->sgte;
    }
    return 0;
}
```

Este programa muestra:

a) 1; 2; 3;
b) 3; 1; 2;
c) 1; 3; 2;
d) 3; 1; 2
e) 2;1; 3;
f) 2; 3; 1;
g) Nada
h) No compila JUSTIFICAR.

3)

```
void mete(Nodo *&raiz, int elemento)
{
    Nodo *aux = raiz;
    if (!raiz)
    {
        raiz = new Nodo();
        raiz-> info = elemento;
        raiz ->sgte = NULL;
    }
    else
    {
        while (aux->sgte)
            aux= aux->sgte;
        aux->sgte = new Nodo();
        aux->sgte->info = elemento;
        aux->sgte->sgte = NULL;
    }
    return;
}
```

```
int saca(Nodo *&raiz)
{
    Nodo *aux; int x;
    if (!raiz)
        return 0;
    aux = raiz;
    x = aux->info;
    raiz = raiz->sgte;
    delete aux;
    return x;
}
```

Los subps “mete” y “saca” podrían implementar:

a) Push y Pop de una pila
b) Queue y Unqueue de una cola
c) Ninguno de los anteriores.

Puede asumir que el tipo “Nodo” es el declarado en el punto 2.

UTN FRBA - Algoritmos y Estructura de Datos – Comisión Ing. Pablo Mendez

Curso: Apellido y Nombres: Legajo: Fecha:

4)

```
int main()
{
    int *p; int x=10;
    p = new int[10];
    p[0] = 20;
    p[1] = 21;
    p[2] = *p;
    p[3] = *(p+1);
    p[4] = *p + *(p+1);
    cout << *(p+4) << endl;
    return 0;
}
```

Este programa muestra:

a) 41
b) 40
c) 42
d) 0
e) Una dirección de memoria.
f) Compila pero tiene comportamiento indeterminado. JUSTIFICAR.
g) No compila. JUSTIFICAR.

5)

```
int main()
{
    FILE *f;
    int x;
    int y = 200;
    f = fopen("prueba.laquequieras","wb");
    int vec[] = {91,92,93,94,95,96,97,98,99,100};
    fwrite(vec,sizeof(int),10,f);
    fclose(f);
    f = fopen("prueba.laquequieras","rb+");
    fseek(f, (-5)*sizeof(int),SEEK_END);
    fread(&x,sizeof(int),1,f);
    fseek(f, 0,SEEK_CUR);
    fwrite(&y,sizeof(int),1,f);
    fseek(f, 0,SEEK_SET);
    fread(&x,sizeof(int),1,f);
    while(!feof(f))
    {
        cout << x << ";";
        fread(&x,sizeof(int),1,f);
    }
    fclose(f);
    return 0;
}
```

Este programa muestra:

a) 91;92;93;
b) 91;92;93;94;95;96;97;98;99;200;
c) 91;92;93;94;95;96;97;98;200;100;
d) 91;92;93;94;95;96;97;200;99;100;
e) 91;92;93;94;95;96;200;98;99;100;
f) 91;92;93;94;95;200;97;98;99;100;
g) 91;92;555;94;95;96;97;98;99;100;
h) 91;92;93;555;94;95;96;97;98;99;100;
i) No compila, JUSTIFICAR.

6)

```
void doThat(int *&p1, int *p2)
{
    p1 = new int();
    *p1 = *p2;
    return;
}

int main()
{
    int x;
    x = 30;
    int *p;
    x++;
    doThat(p, &x);
    cout << *p;
    delete p;
    return 0;
}
```

Este programa muestra:

a) 30
b) 31
c) 0
d) NULL
e) Una dirección de memoria
f) No compila JUSTIFICAR.

7) Declare los nodos necesarios para implementar una lista de caracteres que contenga una sublista de números enteros. **(1 pto).**

8) Implemente un subprograma que reciba un nro. y busque si existe. En caso que exista debe devover el caracter de la lista a la que pertenece. Si no existe puede devolver el carácter '0' **(3 ptos).**

char buscarNro(NodoLetra *lista, int nro);