

Forum:[Hack Forums](#)

Thread: [Just playin with pointers \(Dynamically Allocating a structure of ints?\)](#)

aeiro - August 17, 2018, 7:26 pm

OP

Spoiler ([Click to View](#))

PHP Code

```
<?php
#include "stdafx.h"
#include <conio.h>
struct _test
{
    int i;
    int j;

};

typedef _test* pTest;
#define Make(x, y, z) (*((x*)(y)+z))

pTest funt(int prev, int i, pTest ppTest)
{

    pTest test = (pTest)malloc((sizeof(_test)*i));
    for (int j = 0; j <= 5; j++)
        test[j] = ppTest[j];
    //test->i = 100;

    delete(ppTest);

    return test;
}

int main()
{
    pTest test = (pTest)malloc((sizeof(_test)*5));

    Make(uintptr_t, test, 0) = 0x65;
    /*(uintptr_t*)(test) = 5;
    Make(int, test, 1) = 6;
    /*((BYTE*)(test)+4) = 6;
    Make(uintptr_t, test, 1) = 2;
    /*((uintptr_t*)(test)+ 1) = 2;
    Make(int, test, 3) = 3;
    /*((char*)(test)+12)) = 3;

    test = funt(5, 200, test);

    for (int i = 0; i <= 400; i++)
```

```

{
    Make(int, test, i) = i;

}

_test test1 = *(test + 1);
_test test0 = *test;
for (int i = 0; i <= 400; i++)
    std::cout << Make(int, test, i) << std::endl;
_test test2 = test[2];
_getch();

    return 0;
}

```

UPDATE****

Spoiler ([Click to View](#))

PHP Code

```

<?php
#include "stdafx.h"
#include <conio.h>
struct _test
{
    int i;
    int j;

};

typedef _test* pTest;
#define Make(x, y, z) (((x*)(y)+z))

void Instant(int size, pTest &ppTest)
{

    for (int i = 1; i < size; i++)
    {
        Make(int,ppTest,i) = NULL;
    }
}

void Shrink(int prev,int& i, pTest &ppTest, int pad)
{
    pTest tmptest = (pTest)malloc((sizeof(_test)*prev));

    int k = 0;
    for (int j = 0; j <= prev; j++)
    {
        _test tmp = ppTest[j];
    }
}

```

```

//if ((j % 2) == 0 ) {
if (Make(int, &tmp, 0) > NULL)
{
/* if (j == 0)*/
Make(uintptr_t*, tmpTest, k) = Make(uintptr_t*, ppTest, j);
//else
// Make(uintptr_t*, tmpTest, k) = Make(uintptr_t*, ppTest, j);
k++;
}
//}
}

```

```

ppTest = (pTest)malloc((sizeof(_test)*k + pad));

```

```

Instant(k + pad, ppTest);
for (int j = 0; j < k; j++)
{

```

```

ppTest[j] = tmpTest[j];
//Make(uintptr_t*, ppTest, j
}
free(tmpTest);
i = k;

```

```

}
pTest Alloc(int prev,int i, pTest &ppTest)
{
pTest test = (pTest)malloc((sizeof(_test)*i));
Instant(i, test);
for (int j = 0; j <= prev; j++)
test[j] = ppTest[j];
free(ppTest);
return test;
}

```

```

int main()
{
pTest test = (pTest)malloc((sizeof(_test)*5));
test = Alloc(5,25, test);
Make(uintptr_t, test, 0) = 0x65;
Make(int, test, 1) = 6;
Make(uintptr_t, test, 1) = 2;
Make(int, test, 3) = 3;
Make(uintptr_t, test, 2) = 4;
Make(int, test, 5) = 5;
Make(uintptr_t, test, 3) =6;
Make(int, test, 7) = 7;
Make(uintptr_t, test, 4) = 8;
Make(int, test, 9) = 9;

```

```

pTest testies = new _test();

```

```

testies->i = 69;

```

```
testies->j = 96;
```

```
*&*&*Make(uintptr_t*, test, 1) = Make(uintptr_t*, testies, 0);  
Make(uintptr_t, test, 5) = Make(uintptr_t, testies, 0);  
Make(uintptr_t, test, 6) = Make(uintptr_t, testies, 0);  
delete(testies);
```

```
int nSize = NULL;  
test = Alloc(25, 100, test);  
_test test12 = *(test + 1);  
Shrink(100, nSize, test, 10);  
_test test1 = *(test + 1);  
_test test2 = *(test + 2);  
  
Shrink(nSize, nSize, test, 10);
```

```
_test test0 = *test;
```

```
for (int i = 0; i < nSize; i++)  
printf("Index:%d %d,%d\n", i, Make(uintptr_t, test, i), Make(int, &test[i], 1));
```

```
//_test test2 = test[2];  
_getch();
```

```
    return 0;  
}
```

S'more****

PHP Code

<?php

#include "stdafx.h"

#include <conio.h>

```
int Tampon = 10;
```

```
struct _test
```

```
{  
    int i;  
    int j;  
    int k;  
    int l;  
};
```

```
typedef _test* pTest;
```

```
#define Make(x, y, z) *((x*)(y)+z))
```

```
void Instant(int size, pTest &ppTest)
```

```
{
```

```

    for (int i = 0; i < size; i++)
    {
        Make(int,ppTest,i) = NULL;
    }
}
void Swap(pTest oldT, int oldSize, pTest &newT, int newSize)
{
    newT = (pTest)malloc((sizeof(_test)*newSize));

    for (int i = 0; i < oldSize; i++)
        newT[i] = oldT[i];

}
void Shrink(int prev,int& i, pTest &ppTest)
{
    pTest tmpTest = (pTest)malloc((sizeof(_test)*prev));

    int k = 0;
    for (int j = 0; j < prev; j++)
    {
        _test tmp = ppTest[j];

        if (Make(int, &tmp, 0) > NULL)
        {
            Make(_test, tmpTest, k) = Make(_test, ppTest, j);
            k++;
        }
    }

    ppTest = (pTest)malloc((sizeof(_test)*k));

    Instant(k, ppTest);
    for (int j = 0; j < k; j++)
    {
        ppTest[j] = tmpTest[j];
    }
    free(tmpTest);
    i = k;
}
void Alloc(int prev, int i, pTest &ppTest)
{
    pTest test = (pTest)malloc((sizeof(_test)*i));
    Instant((i*sizeof(_test))/4, test);
    if (prev)
    {
        for (int j = 0; j < prev; j++)
            test[j] = ppTest[j];
    }
    ppTest = test;
}

```

```

}

int main()
{
    pTest test;
    int nSize = 150;
    Alloc(0, nSize, test);
    int sz = (nSize * sizeof(_test)) / 4;
    for (int i = (110 * sizeof(_test)) / 4; i < sz; i++)
        Make(int, test, i) = (i)+1;


    Shrink(nSize, nSize, test);


    pTest swapTest;
    Swap(test, nSize, swapTest, nSize + Tampon);
    int newSize = 0;
    Shrink(nSize + Tampon, newSize, swapTest);
    Alloc(0, newSize, test);
    Swap(swapTest, newSize, test, newSize);


    for (int i = 0; i < nSize; i++)
        printf("Index:%d %d,%d,%d,%d\n", i, Make(int, &test[i], 0), Make(int, &test[i], 1), Make(int, &test
[i], 2), Make(int, &test[i], 3));

    _getch();

    return 0;
}

```

NoK17 - August 18, 2018, 2:22 am

PHP Code

```

<?php
#include "stdafx.h"
#include <conio.h>
struct _test
{
    ...
};

...

pTest funt(int prev, int i, pTest ppTest)
{
    ...
}

int main()

```

```

{
....
}

_test test1 = *(test + 1);
_test test0 = *test;
for (int i = 0; i <= 400; i++)
std::cout << Make(int, test, i) << std::endl;
_test test2 = test[2];
_getch();

return 0;
}

```

[/quote]

Ugh, you're one of those people that put your brackets on a different line.
I wanted to say this was really cool, but then i just saw the brackets :(

Edit: Will compile gimme sec.
Edit2: stdafx.h msvc i luv u
Edit3: its return EXIT_SUCCESS (1) not 0..

-Bracket Jesus, NoK

Dostoevsky - August 18, 2018, 2:35 am

(Aug 18, 2018 02:22 AM)NoK17 Wrote: [\[CITE\]](#) Edit3: its return EXIT_SUCCESS (1) not 0..
EXIT_SUCCESS is a global constant with the value of 0 in basically every implementation.

My 2 cents would be:

Don't bother explicitly returning 0 at the end of main, leave it omitted.

It's been part of the language standard for both C and C++ to implicitly return 0 at the end of main's control flow for years. It was common practice in compilers even before it was standardised behaviour.

NoK17 - August 18, 2018, 4:12 am

(Aug 18, 2018 02:35 AM)Dostoevsky Wrote: [\[CITE\]](#) EXIT_SUCCESS is a global constant with the value of 0 in basically every implementation.
I cannot express my embarrassment :P

Gorzoid - August 18, 2018, 5:03 am

don't free malloc pointers with delete that is for pointers allocated with new only. You should replace delete with free.

aeiro - August 18, 2018, 6:29 am

(Aug 18, 2018 05:03 AM)Gorzoid Wrote: [\[CITE\]](#) don't free malloc pointers with delete that is for pointers allocated with new only. You should replace delete with free.
Yeah i ran into problems with this last night amd figured it out. Thank you.

(Aug 18, 2018 02:22 AM)NoK17 Wrote: [\[CITE\]](#) PHP Code
<?php
#include "stdafx.h"
#include <conio.h>

```

struct _test
{
...
};

...

pTest funt(int prev, int i, pTest ppTest)
{
...
}

int main()
{
....
}

_test test1 = *(test + 1);
_test test0 = *test;
for (int i = 0; i <= 400; i++)
std::cout << Make(int, test, i) << std::endl;
_test test2 = test[2];
_getch();

return 0;
}

```

(Aug 18, 2018 02:22 AM)NoK17 Wrote: [\[CITE\]](#) Ugh, you're one of those people that put your brackets on a different line.

I wanted to say this was really cool, but then i just saw the brackets :(

Edit: Will compile gimme sec.

Edit2: stdafx.h msvc i luv u

Edit3: its return EXIT_SUCCESS (1) not 0..

-Bracket Jesus, NoK"This code is so cool! I know nothing about it! Oh wait no i just got triggered by bracket placement."

You sound dumb.

NoK17 - August 18, 2018, 8:42 am

(Aug 18, 2018 06:29 AM)aeiro Wrote: [\[CITE\]](#) Yeah i ran into problems with this last night amd figured it out. Thank you.

"This code is so cool! I know nothing about it! Oh wait no i just got triggered by bracket placement."
You sound dumb.

You can think I'm a retard, Most likely I am, Most likely you are a master programmer.

Most likely I'm fucking joking, who the fuck cares how you place your brackets?

Your code is cool, no disrespect intended.

-Bracket god.

aeiro - August 18, 2018, 8:51 am

(Aug 18, 2018 08:42 AM)NoK17 Wrote: [\[CITE\]](#) You can think I'm a retard, Most likely I am, Most

likely you are a master programmer.

Most likely I'm fucking joking, who the fuck cares how you place your brackets?

Your code is cool, no disrespect intended.

-Bracket god.

I don't think you are a retard just ignorant lmao. Nor did i feel disrespected.

Non the less this is off topic.

Ill be posting more code once i get off work.

NoK17 - August 18, 2018, 8:55 am

(Aug 18, 2018 08:51 AM)aeiro Wrote: [\[CITE\]](#) I don't think you are a retard just ignorant lmao. Nor did i feel disrespected.

Non the less this is off topic.

Ill be posting more code once i get off work.

Yeah, I get that alot man :D

Post more shit (as in, more code, not actual eh shit, clarification), people post too little in this section.

I will be posting too.

aeiro - August 18, 2018, 1:57 pm

UPDATE****

PHP Code

```
<?php
#include "stdafx.h"
#include <conio.h>
struct _test
{
    int i;
    int j;
};

typedef _test* pTest;
#define Make(x, y, z) *((x*)(y)+z)

void Instant(int size, pTest &ppTest)
{
    for (int i = 1; i < size; i++)
    {
        Make(int,ppTest,i) = NULL;
    }
}

void Shrink(int prev,int& i, pTest &ppTest, int pad)
{
    pTest tmpptest = (pTest)malloc((sizeof(_test)*prev));

    int k = 0;
```

```

for (int j = 0; j <= prev; j++)
{
    _test tmp = ppTest[j];

    //if ((j % 2) == 0 ) {
        if (Make(int, &tmp, 0) > NULL)
        {
/*            if (j == 0)*/
                Make(uintptr_t*, tmp, k) = Make(uintptr_t*, ppTest, j);
            //else
            //    Make(uintptr_t*, tmp, k) = Make(uintptr_t*, ppTest, j);
            k++;
        }
    //}
}

ppTest = (pTest)malloc((sizeof(_test)*k + pad));

Instant(k + pad, ppTest);
for (int j = 0; j < k; j++)
{

    ppTest[j] = tmp[j];
    //Make(uintptr_t*, ppTest, j
}
free(tmp);
i = k;

}
pTest Alloc(int prev,int i, pTest &ppTest)
{
    pTest test = (pTest)malloc((sizeof(_test)*i));
    Instant(i, test);
    for (int j = 0; j <= prev; j++)
        test[j] = ppTest[j];
    free(ppTest);
    return test;
}

int main()
{
    pTest test = (pTest)malloc((sizeof(_test)*5));
    test = Alloc(5,25, test);
    Make(uintptr_t, test, 0) = 0x65;
    Make(int, test, 1) = 6;
    Make(uintptr_t, test, 1) = 2;
    Make(int, test, 3) = 3;
    Make(uintptr_t, test, 2) = 4;
    Make(int, test, 5) = 5;
    Make(uintptr_t, test, 3) = 6;
    Make(int, test, 7) = 7;
    Make(uintptr_t, test, 4) = 8;
    Make(int, test, 9) = 9;

```

```

pTest testies = new _test();

testies->i = 69;
testies->j = 96;

*&*&Make(uintptr_t*, test, 1) = Make(uintptr_t*, testies, 0);
Make(uintptr_t, test, 5) = Make(uintptr_t, testies, 0);
Make(uintptr_t, test, 6) = Make(uintptr_t, testies, 0);
delete(testies);

int nSize = NULL;
test = Alloc(25, 100, test);
_test test12 = *(test + 1);
Shrink(100, nSize, test, 10);
_test test1 = *(test + 1);
_test test2 = *(test + 2);

Shrink(nSize, nSize, test, 10);

_test test0 = *test;

for (int i = 0; i < nSize; i++)
    printf("Index:%d %d,%d\n", i, Make(uintptr_t, test, i), Make(int, &test[i], 1));

//_test test2 = test[2];
_getch();

return 0;
}

```

S'more

PHP Code

```

<?php
#include "stdafx.h"
#include <conio.h>

int Tampon = 10;

struct _test
{
    int i;
    int j;
    int k;
    int l;
};

typedef _test* pTest;
#define Make(x, y, z) ((*((x*)(y)+z))

void Instant(int size, pTest &ppTest)
{

```

```

    for (int i = 0; i < size; i++)
    {
        Make(int,ppTest,i) = NULL;
    }
}
void Swap(pTest oldT, int oldSize, pTest &newT, int newSize)
{
    newT = (pTest)malloc((sizeof(_test)*newSize));

    for (int i = 0; i < oldSize; i++)
        newT[i] = oldT[i];

}
void Shrink(int prev,int& i, pTest &ppTest)
{
    pTest tmpTest = (pTest)malloc((sizeof(_test)*prev));

    int k = 0;
    for (int j = 0; j < prev; j++)
    {
        _test tmp = ppTest[j];

        if (Make(int, &tmp, 0) > NULL)
        {
            Make(_test, tmpTest, k) = Make(_test, ppTest, j);
            k++;
        }
    }

    ppTest = (pTest)malloc((sizeof(_test)*k));

    Instant(k, ppTest);
    for (int j = 0; j < k; j++)
    {
        ppTest[j] = tmpTest[j];
    }
    free(tmpTest);
    i = k;
}
void Alloc(int prev, int i, pTest &ppTest)
{
    pTest test = (pTest)malloc((sizeof(_test)*i));
    Instant((i*sizeof(_test))/4, test);
    if (prev)
    {
        for (int j = 0; j < prev; j++)
            test[j] = ppTest[j];
    }
    ppTest = test;
}

```

```
}
```

```
int main()
```

```
{
```

```
    pTest test;
```

```
    int nSize = 150;
```

```
    Alloc(0, nSize, test);
```

```
    int sz = (nSize * sizeof(_test)) / 4;
```

```
    for (int i = (110 * sizeof(_test)) / 4; i < sz; i++)
```

```
        Make(int, test, i) = (i)+1;
```

```
    Shrink(nSize, nSize, test);
```

```
    pTest swapTest;
```

```
    Swap(test, nSize, swapTest, nSize + Tampon);
```

```
    int newSize = 0;
```

```
    Shrink(nSize + Tampon, newSize, swapTest);
```

```
    Alloc(0, newSize, test);
```

```
    Swap(swapTest, newSize, test, newSize);
```

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

```
        printf("Index:%d %d,%d,%d,%d\n", i, Make(int, &test[i], 0), Make(int, &test[i], 1), Make(int, &test[i], 2), Make(int, &test[i], 3));
```

```
    _getch();
```

```
    return 0;
```

```
}
```