

PB071: pb071_vnitro_12hod

Jméno a příjmení – pište do okénka	UČO	Číslo zadání
		1

Více odpovědí může být správných. Za zcela správně zodpovězenou otázku máte 2 body, lze získat i část bodů za část správných odpovědí. Za každou chybnou odpověď je -1 bod.

1

```
#include <stdlib.h>
#include <stdio.h>
void foo(int** array) {
    *array = malloc(40);
    array = NULL;
}
int main() {
    int* array = NULL;
    foo(&array);
    if (array != NULL) printf("Hello");
    else printf("World");
    free(array);
    return 0;
}
```

Předpokládejte, že alokace malloc(40) proběhne v pořádku. Výše uvedený program:

- A způsobí memory leak o velikosti 40*sizeof(char)
- B vypíše Hello
- C žádná z ostatních odpovědí není správná
- D vypíše World
- E nezpůsobí žádný memory leak

2

```
void foo(int a) {
    for (int i = a; i < 10; i++) {
        if (i > 8) break;
        if (i > 7) continue;
        printf("%d ", i);
    }
}
```

Zavolání funkce foo(3):

- A žádná z ostatních odpovědí není správná
- B vypíše na standardní výstup 3 6 7 8 9 10
- C Způsobí zacyklení programu
- D vypíše na standardní výstup 3 4 5 6 7
- E vypíše na standardní výstup 3 4 5 6 7 8

3

```
#include <stdio.h>
struct X { int value; };
void foo(struct X* a1, struct X a2) {
    a1->value += a2.value;
    a2.value = a1->value;
}

int main() {
    struct X a1 = {1};
    struct X a2 = {2};
    foo(&a1, a2);
    printf("%d %d", a1.value, a2.value);
    return 0;
}
```

Výstupem tohoto programu bude:

- A 3 3
- B 3 2
- C program nelze přeložit
- D žádná z ostatních odpovědí není správná
- E 1 2

```
4 #include <stdio.h>
#include <stdlib.h>
#include <string.h>
typedef struct _item {
    char value1;
    char value2[10];
    char* value3;
} item_t;
void foo() {
    item_t item1 = {'1', "A", NULL};
    item_t item2 = {'2', "B", NULL};
    item1.value3 = malloc(10 * sizeof(char));
    item2.value3 = malloc(10 * sizeof(char));
    strcpy(item1.value3, "X");
    strcpy(item2.value3, "Y");
    printf("%c%s%s", item1.value1, item1.value2, item1.value3);
    printf("%c%s%s", item2.value1, item2.value2, item2.value3);
    item2 = item1;
    strcpy(item1.value3, "Y");
    printf("%c%s%s", item1.value1, item1.value2, item1.value3);
    printf("%c%s%s", item2.value1, item2.value2, item2.value3);
}
int main() {
    foo();
    return 0;
}
```

Pro výše uvedený program platí:

- A Vypíše "1AX2BY1AY1AY"
- B Vypíše "1AX2BY1AX2BY"
- C Žádná z ostatních odpovědí není správná
- D Vypíše "1AX2BY1AX1AX"
- E Nelze přeložit
- F Každé zavolání funkce foo() způsobí memory leak

```
5 #include <stdlib.h>
int main() {
    char* array = malloc(10);
    for (char i = 1; i <= 10; i++) array[i] = i;
    *(array - 1) = 1;
    return 0;
}
```

Pro uvedený kód platí:

- A Program nelze přeložit
- B Program zapíše před začátek alokovaného pole
- C Program se zacyklí
- D Program zapíše za konec alokovaného pole
- E Program vyplní všechny položky alokovaného pole
- F Program neprovádí dealokaci veškeré alokované paměti

```
6 #include <stdio.h>
int array[] = {1, 2, 3, 4};
void foo() {
    int* pVal = array;
    printf("%d ", *pVal);
    pVal++;
}
int main(void) {
    foo();foo();foo();
    return 0;
}
```

Výše uvedený program vypíše:

- A 1 1 1
- B nelze předem určit, záleží na aktuálních hodnotách v poli array
- C 1 2 3
- D žádná z ostatních odpovědí není správná
- E nelze přeložit

```
7 #include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main() {
    char* string = malloc(10);
    strcpy(string, "Test");
    int size1 = sizeof(string);
    int size2 = strlen(string);
    int size3 = 0;
    int offset = 0;
    while (string[offset] != 0) { offset++; size3++; }
    return 0;
}
```

Pro obsah proměnných size1, size2 a size3 těsně před ukončením funkce main() platí:

- A Hodnota proměnné size1 bude 10
 - B Hodnota proměnné size3 bude 10
 - C Hodnota proměnné size3 bude 5
 - D Hodnota proměnné size1 bude sizeof(char*)
 - E Hodnota proměnné size2 bude 10
 - F Hodnota proměnné size2 bude 4
-

```
8 #include <stdio.h>
int main() {
    unsigned int flags = 0;
    flags = flags | 0x02;
    flags = flags | 0x05;
    for (unsigned char i = 1; i <= 8; i = i * 2) {
        if (flags & i) printf("%d ", i);
    }
    return 0;
}
```

Výše uvedený program:

- A Žádná z ostatních odpovědí není správná
 - B Vypíše 2 8
 - C Vypíše 1 4
 - D Vypíše 1 2 3 4 5 6 7 8
 - E Vypíše 1 2 4
 - F Vypíše 2 5
-

```
9 int array[] = {1, 2, 3, 4};
int main(void) {
    array = 5;
    return 0;
}
```

Překladač vypsál během překladač výše uvedeného programu následující:

..\main.c: In function 'main':

..\main.c:3:11: error: incompatible types when assigning to type 'int[4]' from type 'int'

Které z uvedených změn kódu umožní provedení překladač bez chyb?

- A deklarovat pole array jako array[4] namísto array[]
 - B dereferencovat proměnnou array před přiřazením: *array = 5;
 - C specifikovat index položky: array[0] = 5;
 - D získat adresu místa obsahující konstantu 5 před přiřazením do array: array = &5;
 - E přetypovat přiřazovanou hodnotu 5 na int[4]: array = (int[4])5;
 - F k vypsání této chyby překladačem nemohlo dojít
-

10

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void foo(int*** array) {
    const int size= 3;
    *array = malloc(sizeof(int*) * size);
    for (int i = 0; i < size; i++) (*array)[i] = malloc(sizeof(int) * (i+1));
}
int main(void) {
    int** myArray;
    foo(&myArray);
    free(myArray);
    return 0;
}
```

Výše uvedený program:

- A** Dynamicky naalokuje pravoúhlé (ve všech rozměrech stejný počet prvků) trojrozměrné pole do ukazatele array
- B** Nelze přeložit
- C** Dynamicky naalokuje nepravoúhlé (v různých rozměrech může mít různý počet prvků) dvojrozměrné pole do ukazatele array
- D** Způsobí memory leak
- E** Dynamicky naalokuje pravoúhlé (ve všech rozměrech stejný počet prvků) dvojrozměrné pole do ukazatele array
- F** Dynamicky naalokuje nepravoúhlé (v různých rozměrech může mít různý počet prvků) trojrozměrné pole do ukazatele array