Zadanie 1 – Správca pamäti

V štandardnej knižnici jazyka C sú pre alokáciu a uvoľnenie pamäti k dispozícii funkcie malloc a free. V tomto zadaní je úlohou implementovať vlastnú verziu alokácie pamäti.

Konkrétnejšie je vašou úlohou je implementovať v programovacom jazyku C nasledovné ŠTYRI funkcie:

- void *memory_alloc(unsigned int size);
- int memory_free(void *valid_ptr);
- int memory_check(void *ptr);
- void memory_init(void *ptr, unsigned int size);

Vo vlastnej implementácii môžete definovať aj iné pomocné funkcie ako vyššie spomenuté, nesmiete však použiť existujúce funkcie malloc a free.

Funkcia **memory_alloc** má poskytovať služby analogické štandardnému malloc. Teda, vstupné parametre sú veľkosť požadovaného súvislého bloku pamäte a funkcia mu vráti: ukazovateľ na úspešne alokovaný kus voľnej pamäte, ktorý sa vyhradil, alebo NULL, keď nie je možné súvislú pamäť požadovanej veľkosť vyhradiť.

Funkcia **memory_free** slúži na uvoľnenie vyhradeného bloku pamäti, podobne ako funkcia free. Funkcia vráti 0, ak sa podarilo (funkcia zbehla úspešne) uvoľniť blok pamäti, inak vráti 1. Môžete predpokladať, že parameter bude vždy platný ukazovateľ, vrátený z predchádzajúcich volaní funkcie **memory_alloc**, ktorý ešte nebol uvoľnený.

Funkcia **memory_check** slúži na skontrolovanie, či parameter (ukazovateľ) je platný ukazovateľ, ktorý bol v nejakom z predchádzajúcich volaní vrátený funkciou **memory_alloc** a zatiaľ nebol uvoľnený funkciou **memory_free**. Funkcia vráti 0, ak je ukazovateľ neplatný, inak vráti 1.

Funkcia **memory_init** slúži na inicializáciu spravovanej voľnej pamäte. Predpokladajte, že funkcia sa volá práve raz pred všetkými inými volaniami **memory_alloc**, **memory_free** a **memory_check**. Viď testovanie nižšie. Ako vstupný parameter funkcie príde blok pamäte, ktorú môžete použiť pre organizovanie a aj pridelenie voľnej pamäte. Vaše funkcie nemôžu používať globálne premenné okrem jednej globálnej premennej na zapamätanie ukazovateľa na pamäť, ktorá vstupuje do funkcie **memory_init**. Ukazovatele, ktoré prideľuje vaša funkcia **memory_alloc** musia byť výhradne z bloku pamäte, ktorá bola pridelená funkcii **memory_init**.

Okrem implementácie samotných funkcií na správu pamäte je potrebné vaše riešenie dôkladne otestovať. Vaše riešenie musí byť 100% korektné. Teda pokiaľ pridelíte pamäť funkciou **memory_alloc**, mala by byť dostupná pre program (nemala by presahovať pôvodný blok, ani prekrývať doteraz pridelenú pamäť) a mali by ste ju úspešne vedieť uvoľniť funkciou **memory_free**. Riešenie, ktoré nespĺňa tieto minimálne požiadavky je hodnotené 0 bodmi. Testovanie implementujte vo funkcii **main** a výsledky testovania dôkladne zdokumentuje. Zamerajte sa na nasledujúce scenáre:

- prideľovanie rovnakých blokov malej veľkosti (veľkosti 8 až 24 bytov) pri použití malých celkových blokov pre správcu pamäte (do 50 bytov, do 100 bytov, do 200 bytov),
- prideľovanie nerovnakých blokov malej veľkosti (náhodné veľkosti 8 až 24 bytov) pri použití malých celkových blokov pre správcu pamäte (do 50 bytov, do 100 bytov, do 200 bytov),
- prideľovanie nerovnakých blokov väčšej veľkosti (veľkosti 500 až 5000 bytov) pri použití väčších celkových blokov pre správcu pamäte (aspoň veľkosti 1000 bytov),
- prideľovanie nerovnakých blokov malých a veľkých veľkostí (veľkosti od 8 bytov do 50 000) pri použití väčších celkových blokov pre správcu pamäte (aspoň veľkosti 1000 bytov).

V testovacích scenároch okrem iného vyhodnoťte koľko % blokov sa vám podarilo alokovať oproti ideálnemu riešeniu (bez vnútornej aj vonkajšej fragmentácie). Teda snažte sa prideľovať bloky až dovtedy, kým v ideálnom riešení nebude veľkosť voľnej pamäte menšia ako najmenší možný blok podľa scenára.

Príklad jednoduchého testu:

```
#include <string.h>

int main()
{
	char region[50];                          //celkový blok pamäte o veľkosti 50 bytov
	memory_init(region, 50);
	char* pointer = (char*) memory_alloc(10);  //alokovaný blok o veľkosti 10 bytov
	if (pointer)
		memset(pointer, 0, 10);
	if (pointer)
		memory_free(pointer);
	return 0;

}
```

Riešenie zadania sa odovzdáva do miesta odovzdania v AIS do stanoveného termínu (oneskorené odovzdanie je prípustné len vo vážnych prípadoch, ako napr. choroba, o možnosti odovzdať zadanie oneskorene rozhodne cvičiaci, príp. aj o bodovej penalizácii). Odovzdáva sa jeden **zip** archív, ktorý obsahuje jeden zdrojový súbor s implementáciou a jeden súbor s dokumentáciou vo formáte **pdf**.

Pri implementácii zachovávajte určité konvencie písania prehľadných programov (pri odovzdávaní povedzte cvičiacemu, aké konvencie ste pri písaní kódu dodržiavali) a zdrojový kód dôkladne okomentujte. Snažte sa, aby to bolo na prvý pohľad pochopiteľné.

Dokumentácia musí obsahovať hlavičku (kto, aké zadanie odovzdáva), stručný opis použitého algoritmu s názornými nákresmi/obrázkami a krátkymi ukážkami zdrojového kódu, vyberajte len kód, na ktorý chcete extra upozorniť. Pri opise sa snažte dbať osobitý dôraz na zdôvodnenie správnosti vášho riešenia – teda dôvody prečo je dobré/správne, spôsob a vyhodnotenie testovania riešenia. Nakoniec musí technická dokumentácia obsahovať odhad výpočtovej (časovej) a priestorovej (pamäťovej) zložitosti vášho algoritmu. Celkovo musí byť cvičiacemu jasné, že viete čo ste spravili, že viete odôvodniť, že to je správne riešenie, a viete aké je to efektívne.

Hodnotenie

Môžete získať 15 bodov, **minimálna požiadavka 6 bodov**.

Jedno zaujímavé vylepšenie štandardného algoritmu je prispôsobiť dĺžku (počet bytov) hlavičky bloku podľa jeho veľkosti. Každé funkčné vylepšenie cvičiaci zohľadní pri bodovaní.

Cvičiaci prideľuje body podľa kvality vypracovania. 8 bodov môžete získať za vlastný funkčný program prideľovania pamäti (**aspoň základnú funkčnosť musí študent preukázať, inak 0 bodov**; metóda implicitných zoznamov najviac 4 body, metóda explicitných zoznamov bez zoznamov blokov voľnej pamäti podľa veľkosti najviac 6 bodov), 3 body za dokumentáciu (bez funkčnej implementácie 0 bodov), 4 body môžete získať za testovanie (treba podrobne uviesť aj v dokumentácii). Body sú ovplyvnené aj prezentáciou cvičiacemu (napr. keď neviete reagovať na otázky vzniká podozrenie, že to **nie je vaša práca, a teda je hodnotená 0 bodov**).