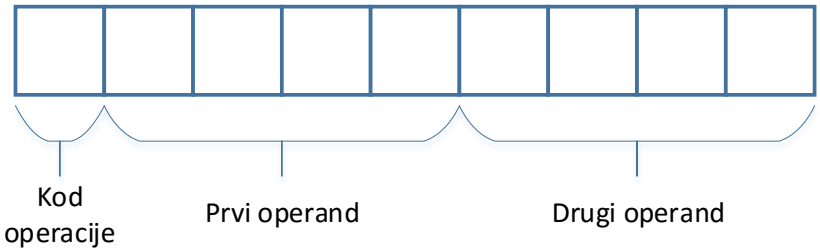
Zadaci za vežbu

Zadatak 1.

U programskom jeziku C/C++ implementirati funkcijiu čija je deklaracija:

```
float* Calculate(char* buffer);
```

Funkcija kao parameter prima pokazivač na niz bajtova. Niz bajtova treba interpretirati kao na slici ispod. Prvi bajt predstavlja kod matematičke operacije (0 - sabiranje, 1 – oduzimanje, 2 – množenje, 3 - deljenje). Naredna četiri bajtra predstavljaju prvi operand - broj u celobrojnom obliku (*integer*), nakon prvog operanda slede naredna četiri bajta koja predstavljaju drugi operand - broj u celobrojnom obliku.



Na osnovu podataka dobijenih u nizu, primeniti odgovarajuću matematičku operaciju na operande. Povratna vrednost funkcije predstavlja pokazivač na broj u pokretnom zarezu (*float*) koji se dobije kao rezultat matematičke operacije.

Napisati test program za ovu funkciju.

Zadatak 2.

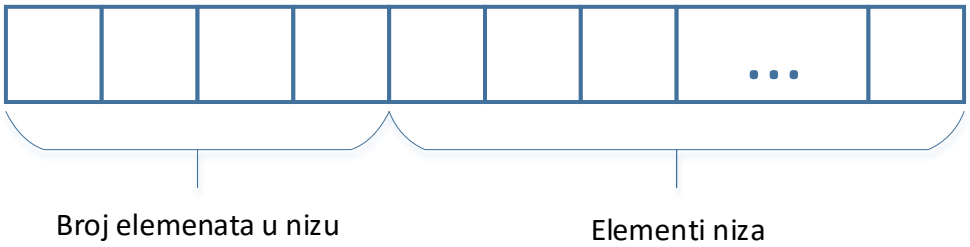
Doraditi prvi zadatak, tako da funkcija interpretira brojeve u *Big endian* formatu. Za konvertovanje brojeva iz *Little endian* u *Big endian* koristiti funkcije *htonl()* i *htnof()* (*host to network long / float*) koje se nalaze u *Winsock2.h* hederu. Za konvertovanje iz *Big endiana* u *Little endian* koristiti funkcije *ntohl()* i *ntohf().*

Zadatak 3.

U programskom jeziku C/C++ implementirati funkcijiu čija je deklaracija:

```
char* CalculateArray(char* buffer);
```

Funkcija kao parameter prima pokazivač na niz bajtova. Niz bajtova treba interpretirati kao na slici ispod. Prva četiri bajta predstavljaju broj elemenata predstojećeg niza u celobrojnom obliku (*integer*). Nakon toga sledi niz bajta koji predstavlja niz brojeva tipa *unsigned char*.



Izračunati sumu brojeva u nizu i prosečan broj niza. Povratna vrednost funkcije je pokazivač na niz od osam bajtova, gde prva četiri bajta predstavljaju prosečan broj niza u *float* format, dok druga četiri bajta predstavljaju sumu brojeva u nizu u *int* formatu.

Napisati test program za ovu funkciju.

Zadatak 4.

U programskom jeziku C/C++ implementirati jednostruko spregnutu listu celobrojnih vrednosti (*integer*). Implementirati sledeće funkcije:

- `Add(int number)` – dodaje broj na kraj liste,
- `Insert(int index, int number)` – dodaje broj u listu na odgovarajuću poziciju,
- `int ElementAt(int index)` – pronalazi broj na odgovarajućoj poziciji u listi,
- `RemoveAt(int index)` – briše broj sa odgovarajuće pozicije u listi,
- `Clear()` - briše sve brojeve iz liste,
- `int[] ToArray()` – kreira niz brojeva na osnovu liste.

Napisati test program.

Zadatak 5.

U programskom jeziku C/C++ implementirati stek (LIFO) celobrojnih vrednosti (*integer*). Implementirati sledeće funkcije:

- `Push(int number)` – stavlja broj na vrh steka,
- `int Pop()` – uklanja broj sa vrha steaka,
- `int Top()` – vraća broj na vrhu steak.

Napisati test program.

Zadatak 6.

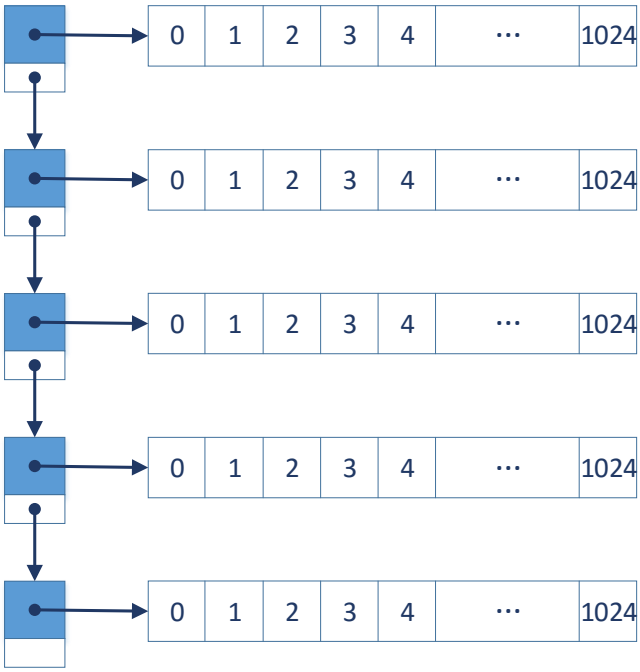
U programskom jeziku C/C++ implementirati red (FIFO) celobrojnih vrednosti (*integer*). Implementirati sledeće funkcije:

- `Enqueue(int number)` – stavlja broj na kraj reda,
- `int Dequeue()` – vraća i uklanja broj sa početka reda.

Napisati test program.

Zadatak 7.

U programskom jeziku C/C++ implementirati *buffer (memory) pool* kao na slici ispod.



Inicijalno, lista sadrži 128 pokazivača na alocirane nizove od 1024 bajta (*buffer*-e) koji se nalaze na *heap*-u. Kada se koristi neki od buffer-a, njegov pokazivač se uklanja iz liste, nakon upotrebe pokazivač na buffer se vraća u listu. Implementirati sledeće funkcije:

- `char* GetNewBufferFromPool()` – daje pokazivač na buffer koji je spreman za korištenje,
- `ReturnBufferToPoll(char* buffer)` – vraća buffer u listu

Napisati test program.

Zadatak 8.

U programskom jeziku C/C++ inicijalizovati niz i listu (iz zadatka 4.) za 1 000 000 celobrojnih vrednosti. Napuniti niz i listu brojevima od 1 000 000 do 1 u obrnutom redosledu, tako da se na indeksu 0 nalazi vrednost 1 000 000, na indeksu 1 nalazi vrednost 999 999…

Sortirati niz i listu koristeći bubble sort algoritam. Prodiskutovati vreme sortiranja niza i liste.

Zadatak 9.

U programskom jeziku C/C++ napraviti strukturu BigStruct koja će simulirati veliku strukturu u kojoj je samo jedan član tipa float od interesa.

```
struct BigStruct
{
	char data1[500];
	float data2;
	char data3[500];
};
```

Napraviti i inicijalizovati nasumičnim brojevima niz pomenute strukture od 5 000 000 elemenata. Napraviti i inicijalizovati nasumičnim brojevima niz float-ova od 5 000 000 elemenata. Izmeriti vreme koje je potrebno da se saberu svi float-ovi iz prvog niza, pa uporediti sa vremenom koje je potrebno za sabiranje svih float-ova iz drugog niza. Prodiskutovati dobijene rezultate za Debug i Release konfiguraciju build-a.