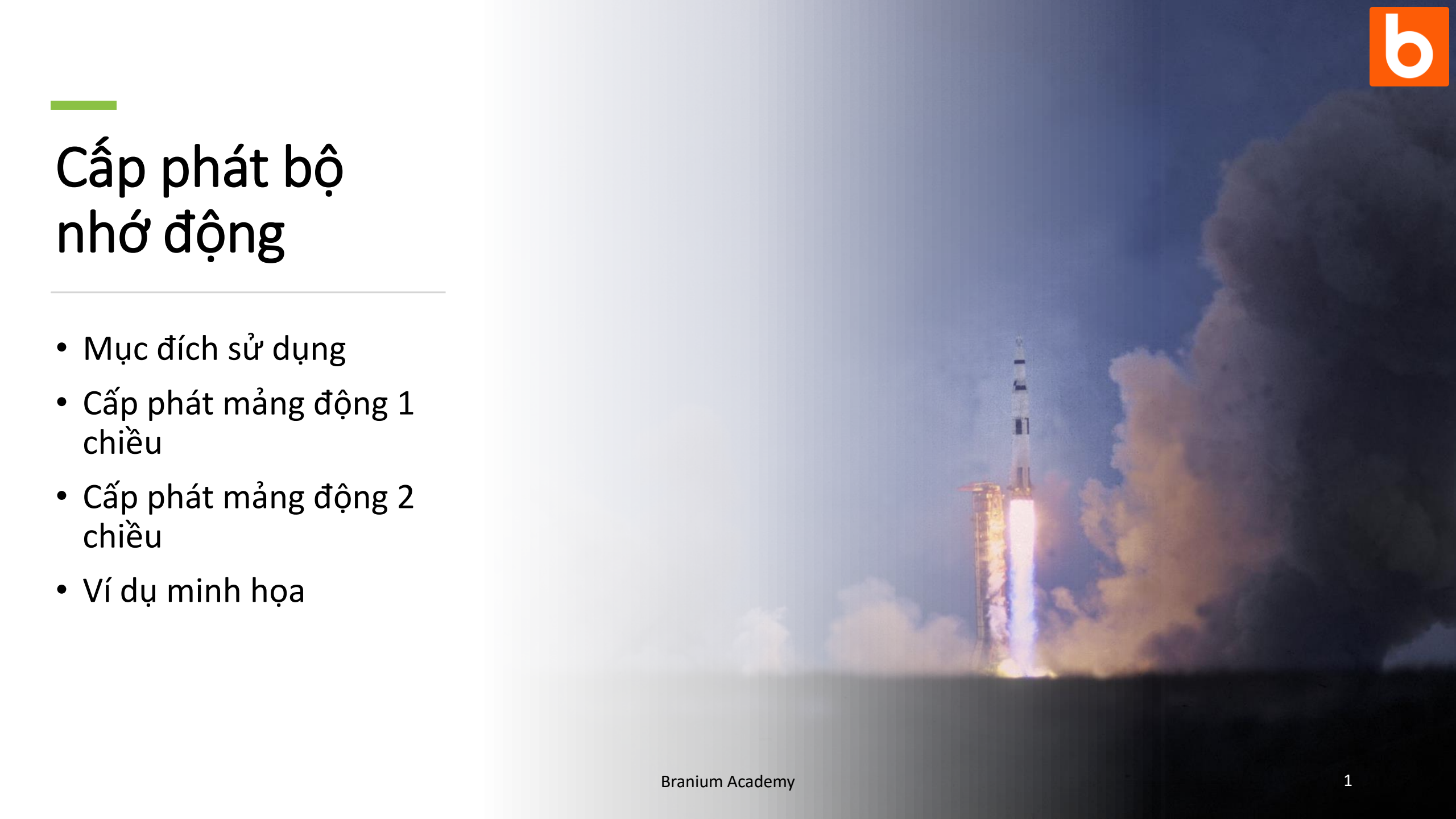
Cấp phát bộ nhớ động

- Mục đích sử dụng
- Cấp phát mảng động 1 chiều
- Cấp phát mảng động 2 chiều
- Ví dụ minh họa



Mục đích sử dụng

- Mảng cấp phát tĩnh thường có nhược điểm là thừa hoặc thiếu các phần tử khi sử dụng
- Thừa trong trường hợp cấp phát quá nhiều mà chỉ sử dụng một phần
- Thiếu trong trường hợp cấp phát ít hơn thực tế cần sử dụng
- Như vậy khi muốn tối ưu bộ nhớ, số lượng phần tử của mảng chỉ được biết tới khi chạy chương trình, ta sẽ sử dụng cấp phát động

Cấp phát mảng động 1 chiều

Cú pháp tổng quát:

con_trỏ = (kiểu)malloc(số_p.tử*sizeof(kiểu));*

- Kiểu: bất kì kiểu nào đó hợp lệ trong ngôn ngữ lập trình C
- Con trỏ: tên con trỏ cần cấp phát
- (kiểu*) dùng để ép kiểu con trỏ vì hàm malloc trả về con trỏ void*
- Malloc là hàm cấp phát bộ nhớ động nằm trong thư viện <stdlib.h>
- Số p.tử là số phần tử cần cấp phát. Mỗi phần tử sẽ có kích thước lấy qua toán tử sizeof(kiểu)

Cấp phát mảng động 1 chiều

Cú pháp tổng quát:
con_trỏ = (kiểu)malloc(số_p.tử*sizeof(kiểu));*

- Bản chất bên trong () của malloc là tổng kích thước bộ nhớ cần cấp phát cho mảng
- Ví dụ: `aPtr = (int*)malloc(n * sizeof(int));`

Sau khi sử dụng mảng ta chủ động thu hồi bộ nhớ:
free(tên_con_trỏ_cấp_phát_động);

Ví dụ:

```
free(aPtr); // giải phóng bộ nhớ đã cấp phát cho aPtr
```

Ví dụ minh họa

- Ví dụ sau minh họa cấp phát động cho mảng int:

```
#include <stdio.h>
#include <stdlib.h>

// ham nguyen mau
void assignArrayElements(int* const a, const size_t n);
void showArray(const int* const a, const size_t n);

int main() {
    size_t n;
    int* aPtr;
    puts("Nhap so phan tu cua mang: ");
    scanf("%u", &n);
    // cap phat bo nho cho con tro aPtr
    aPtr = (int*)malloc(n * sizeof(int));
    assignArrayElements(aPtr, n);
    puts("\nMang vua nhap la: ");
    showArray(aPtr, n);
    // thu hoi bo nho
    free(aPtr);
```

Cấp phát mảng động 2 chiều

Ta dùng con trỏ trỏ đến con trỏ (**) để thao tác.
Cú pháp tổng quát gồm hai bước:

- Bước 1: cấp phát số hàng:
  *arr = (kiểu**)malloc(r*sizeof(kiểu*));*
- Bước 2: cấp phát số cột:

```
for(i = 0; i < r; i++) {
    arr[i] = (kiểu*)malloc(c*sizeof(kiểu));
}
```

- arr là tên con trỏ kép
- r là số hàng
- c là số cột

Giải phóng bộ nhớ cấp phát mảng động 2 chiều

Cú pháp tổng quát gồm hai bước:

- Bước 1: giải phóng từng hàng:

```
for(i = 0; i < r; i++) {
    free(arr[i]);
}
```

- Bước 2: giải phóng con trỏ kép:

```
free(arr);
```

- *Ví dụ:*

```
// buoc 1
for (i = 0; i < row; i++) {
      free(arr[i]);
}
// buoc 2
free(arr);
```

Ví dụ minh họa

Ví dụ sau cấp phát mảng động hai chiều kiểu int:

```c
#include <stdio.h>
#include <stdlib.h>

void fill2DArrElements(int** arr, const size_t row, const size_t col);
void show2DArrElements(const int** arr, const size_t row, const size_t col);

int main() {
    int** arr;
    size_t row = 3; // so hang
    size_t col = 4; // so cot
    // buoc 1
    arr = (int**)malloc(row * sizeof(int*));
    // buoc 2
    size_t i;
    for (i = 0; i < row; i++) {
        arr[i] = (int*)malloc(col * sizeof(int));
    }
    // su dung mang hai chieu cap phat dong
    fill2DArrElements(arr, row, col);
    show2DArrElements(arr, row, col);

    // giai phong bo nho sau khi su dung
    for (i = 0; i < row; i++) {
        free(arr[i]); // giai phong hang
    }
    free(arr); // giai phong con tro kep
```

Tiếp theo

Kí tự và chuỗi kí tự