/ /

Memory allocation :-

ماهو ال heap ؟ هي المكان اللذى يتبع ال bss ويحصله
grow up wards وممكن يحصله Shrink
ومن حاجة اسمها ال program break دة بيقولى نهاية ال heap
بتاعتى فين حالياً أول هما ببدأ خالص بيكون واقف عند ال end
بتاعت ال bss (&end) ودة بيعرف البرنامج
نهاية الـ heap بتاعى فين أو نهاية ال pages اللى حصلها
allocate لل heap فين دة مش بيفصل بين ال allocated
وال non allocated او ال freed دة بيقولى آخر ال
physical memory اللى allocated لل heap مش اللى
حصلها allocation بال malloc

طب لو عايزة اعمل adjust لل program break دى أعملها
أزاى ؟ عايزة مثلا ازود pages علشان اعرف اعمل فيها malloc
أو أقلل ؟
هيحصل بـ system call بـ يهيات brk() , sbrk()
لو دورت عليها فى ال man page مكتوب
Change data segment size
من اول ال data. لحد ال heap هو معتبر دة data segment

```
int brk (Void * addr)
```

بديها النهايه فين (أنا بعرف ال addresses من ال mapped file)

```
Void * sbrk ( intptr_t increment)
```

بديها هزود بـ قد ايه
لو ندهتها بـ 0 هترجع لل ptr ال addr الحالى يعنى ال
Current Program break

/ /

brk() and sbrk() : Changes the location of program break which defines end of process data segment (heap)

* ومش معنى أنى زودت الـ Program break الفكرة
حطها allocation & بالعكس اسلوبك لا ابدا اعمل access
الـ kernel هيبدأ تعمل allocate بعد كذا نوع من انواع
الـ optimization

ماهى الـ boundaries بتاعت الـ Program break ؟

الـ lower limit هو الـ end بتاعت الـ bss
والا لو رجعت قبلها هتكون بعمل free لل bss والـ data
والدنيا هتبوظ

والـ upper limit هى مش الـ top بتاع الـ Stack
الجزء بين الـ Stack والـ heap فية
Shared libraries و memory mapped

بالتالى الـ upper limit بيتحدد بكل

- Stack
- Shared libraries
- Memory mapped
- limitations on process data segment

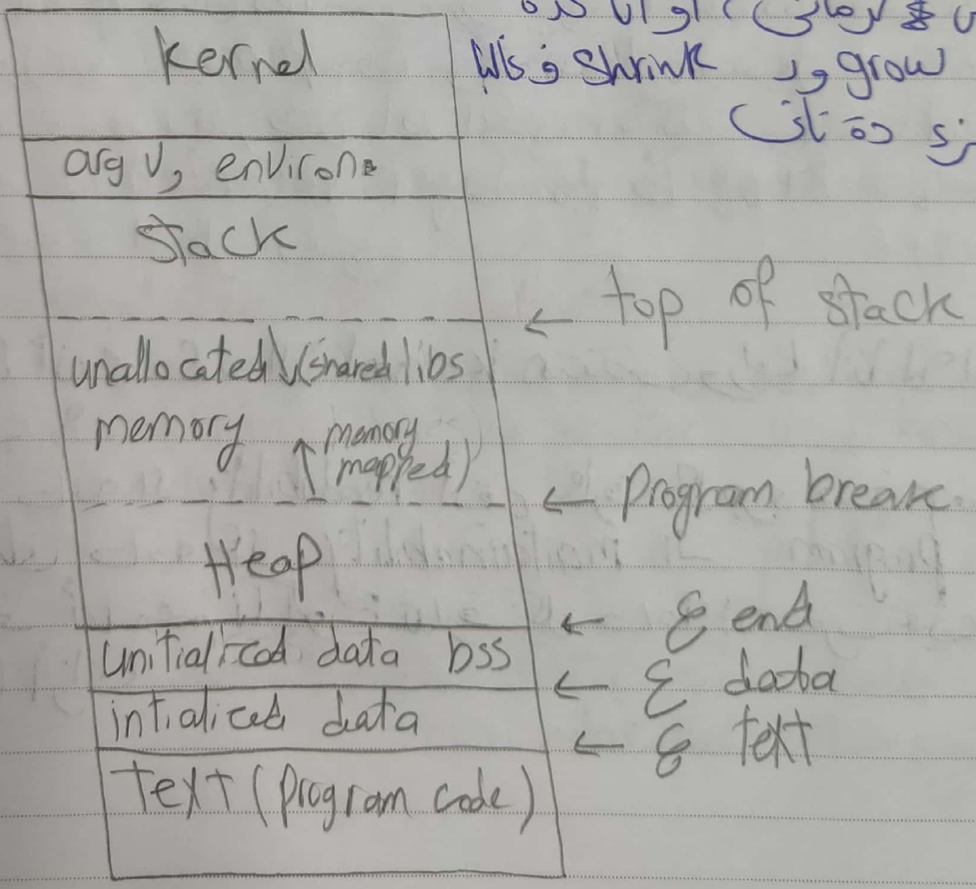
انا بشوف الـ shared library أنا بنيتها mapping ع
مكان الـ Virtual memory بتاعتى

Current program break → sbrk(0)

Malloc() and free ()

الـ malloc() زى ما اناعارفين بتحجز الـ size اللى
أنا عايزه بالـ bytes وممكن تعمل adjust للـ program break
طب ازاى ؟ هى هترجع لـ call الـ brk من جواها
اصلاً

الـ free() هتعمل deallocate للميمورى وهى ممكن مش
لازم ومش دايما بتعمل adjust للـ program break طب ليه ممكن؟
يعنى مثلاً لو أنا كنت محوله memory allocation فى نص
الـ page وفى قبلها وبعدها حاجات كتيرمش هـ free الـ page
كلها واضيع الحاجات دى لكن مثلاً" لو أنا فى أول الـ page
وروحت عملت free ممكن وقتها اعمل كدة لأن الـ page كلها
مبقتش لزمانى ، أو أنا كدة
كدة بـ grow وبـ shrink فـ كلها
هحتاج الجزء دة تانى



/ /

ف أنا مش كل ما اعمل free هعمل call لل brk
أولاً لأن دة headach كل شوية هروح انا هنا وال system calls
عارفين أن ليها over head عالى

ثانيا ممكن تكون الـ freed memory في نص الـ Page

ثالث حاجة أن في Propability أن احتاج الجزء دة تاني

طب الـ process لما بتخلص ايه بيحصل الـ allocated memory
بتاعتها الـ kernel هتاخد الـ pages اللي انا عاملها allocate
ترجعها الـ pool بتاعها free عشان لما تيجي process
جديدة تعرف تستخدمها تاني

ولوقت اللي هيعمل free الـ physical memory الـ kernel بيروح
تـ allocate الـ process وكل الحاجة بتاعتها وتحط عليها حبة
restrictions كدة معدش حد يروح يبوظ المكان دة كأنها بتعمله mark
المحجوز للـ process دى ولما بتخلص بتشيل الـ restrictions
دى ف أى حد يقدر ياخد المكان دة بـ overwrite عليه
لأنه no longer reserved فالأحسن لكن هو مش
هتحطه كله بـ اصفار أو زيرو

طب لما أن كدة كدة الـ kernel هتنضف ورايا أنا ليه اعمل free
بـ ايدى ؟
أول حاجة الـ readability أبقى عارف كل جزء مستخدم فين ولحد
امتى تانى حاجة الـ maitainablity الـ program عال
يكبر ف أنا مهم ليا أكون عارف كل حاجة اتحجزت فين وأقدر
اصلحها أزاى

تالت حاجة واهم حاجة ال long running Programs
زي ال Daemons مثلاً وال Shell دي شغاله وقت
طويل جدا ف ممكن من كتر ال allocation ال memory تخلص
ومنقدرش نكمل عشان كدة ال free مهم جداً هنا

وكمان عشان اتفادى ال Memory leaks :
ودة مثلاً أنت عملت allocate لجزء في الميموري والجزء دة
ميقاش reachable بعدها يا ضيعت ال pointer الي كان
بيشاور عليه بالتالي هو محجوز ومش هعرف اعمله free خلاص
هواخد مساحة وخلاص

عكس حاجة اسمها memory fragmentation

مثلاً لو عملت malloc لـ 4 bytes بعدها malloc لـ 1 bytes
بعدها malloc لـ 3 bytes ورجعت عملت free
لـ 1 byte دي كذا مرة ويقى مثلاً ال heap كلها
مليانة وأنا بدور على 12 byte على بعض مش لاقي بس موجودين
متفرقين عبارة عن gaps بين ال mallocs دي دة بقى
اسمة memory fragmentation

Implementation of Malloc

أول ما نبدأ خالص مش بيكون عندي pages مخصصها allocation
لل heap ف ال malloc بتبدأ تنادي ال brk() عشان
تحجز pages وبعدكدة تبدأ تحجز المساحة اللي عايزها
وترجع لنا pointer بيشاور عليها ، هل هتحجز المساحة بالظبط
مثلاً مداها 10 bytes ولا هتحجز اكتر ؟ هتحجز اكتر
لأنها هتـ save شوية meta data (data about data)
عن ال memory allocated دي

مثلاً لو أنا حجزت 10 bytes وعملت ليهم free الـ free
هيعرف ازاي اني حجزت 10 مش 20 ؟
من الـ meta data دى اللي هي الـ length

لو جيت بعدها عملت malloc لـ 10 عشان مش هزود
page تانية طالما أنا within الـ page الأولى ولسه
عندي مساحة

طب دلوقت لو أنا مثلاً عندي freed block في النص
هل لما اجي اعمل malloc تاني هدور على اللي في النص
دول ولا مليش دعوة و اجي من اول مكان فاضي بعد آخر حاجة
حجزتها وخلاص ؟

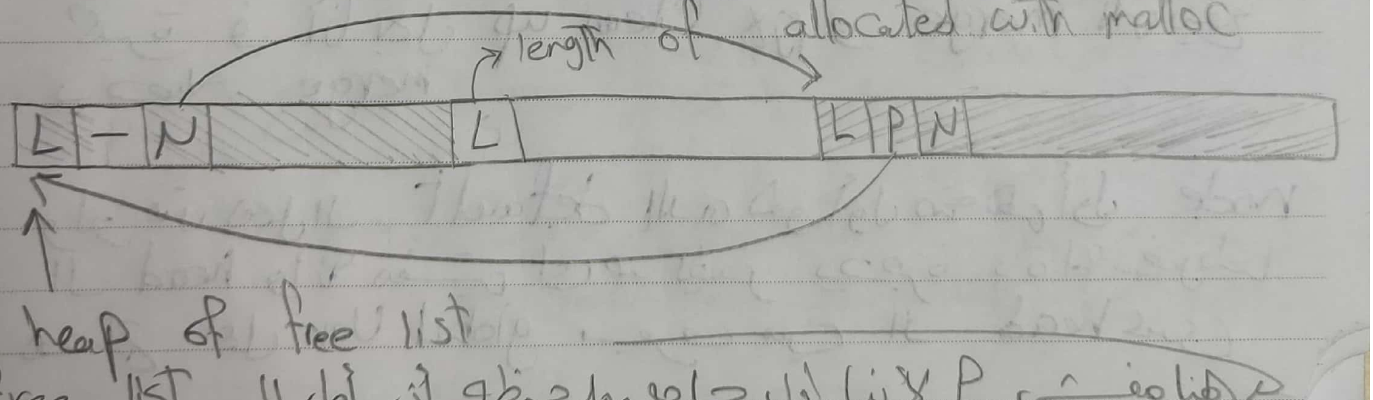
الفكرة ان عندي حاجة اسمها free list فيها كل الحاجات
اللي متاحة من الميموري ومعمولها free ف أنا عندي كذا algorithm
بيدور في الـ free list دى منهم Dest fit و first fit
مثلاً عايزة احجز 20 bytes هروح كـ مثال ادور على 20
لو لقيتهم بالظبط احطهم طب لو لقيت اكبر؟ ممكن اقسم
الـ block دة بحيث احط الـ size right طب لو ملقتش
خالص أو لقيت اقل اعمل ايه؟ هروح انده sbrk() تعمل allocate
هـ لـ pages زيادة اروح ا allocate فيها

أنا الـ ميموري عندي فيها حاجات freed من الملغي وحاجات معمولها allocated طب أنا هـ keep اني اكون tracked بالحاجات الفري دي أزاي؟

هعمل doubly linked list هي دي الـ free list هتعرفلي اماكن الـ memory الـ freed وممكن تزيد وتقل عادي ودة شكل الـ node بتاعتها

Length of block (L)	Pointer to Previous free block (P)	Pointer to next free block (N)	Remaining bytes of free block

يعني هي بتاخد جزء من الـ block دي free كجزء من الـ linked list بتاعتها ودة ذكاء رهيب عشان بدل ماتحمل الـ headache بتاعت اني اشوف مكان لـ linked list مناسب لأنها تكبر وتصغر واقعد اعمل حوارات 8 بستخدمه هو نفسه وبوفر على نفسي ويبقى انظف كدة



وهنا مفيش P لأنها أول حاجة وملحوظة أن أول الـ free list ثابت جوا أول مكان في الـ heap

/ /

هنتعامل معاها معاملة linked list هيا يعني مثلاً لو أنا عايز
block من النص اعمله allocate كده كأني مسحت node
من النص ف هروح افلى next اللى قبلى بيشاور على اللى
ورايا و previous اللى ورايا بيشاور على اللى قبلى
وكده كأنه linked list pure عاديه خالص
دة فى حالة انى هاخد ال block كده على بعضه طب لو ال size
بتاعة اكبر من اللى عايزاه لازم اعمل split
ف هروح اكتب فى الاول فى اول متعمش ال node بس
لازم لازم قبلها اروح من آخر ال node اعمل node
جديدة واعملها insert فى ال free list

أو ← اروح اعمل ال update لل size بتاعت ال node
يعني هيبقى size = size - allocated_size
واروح اقص من آخر ال node ودة هيوفرلى وقت
ال insertion بتاعت ال node

وممكن S نوع من انواع ال optimization لأن لو عندى
كذا node ورا بعض صغيرين اجمعهم فى node واحدة
كبيرة ف أنا بعمل Clean up لازم و merge للى ينفع
يحصله merge

ولو مش عامل ال default اللى هو أول مه هى أول node
ال head وانا مفيش أى حد يقدر يحجزة دة الافضل عليا
هحتاج اعمل global var يعرف مين ال head عندى

وفي عندي calloc () و الفرق بس انها بتعمل allocate وتصفرهم

وفي ال realloc () ودي بتعمل resize لل allocated memory

void *realloc (void * ptr , size_t size);

← بتاخد ال pointer القديم وال size الجديد وترجع ال pointer الجديد

realloc (ptr - Null , size-allocate)
≡ malloc
لواديتها ptr null و size هتتصرف زي ال malloc

realloc (ptr , 0) ≡ free (ptr)
← لواديتها ptr و size zero كأن عملت free

طب لو جيت ازود ال size لو بعدي على طول فى free block هاخده عادي أو آخد بعضه واعمل ال update لل free list
طيب لو مش بعدي free block ؟
هروح انقل ال block كله فى حته تانيه هروح احجز block بال size اللي أنا عايزه وبعد ها اعمل ال copy لل data وفى الآخر اعمل free للقديم ودي من مساوئ ال realloc
(طبعا هعمل ال update لل free list)

لو جيت اقلل ال size هعمل ال split لل block واعمل node جديدة واعملها insert

لازم معلش assign ال pointer بتاعى ال realloc على
طول لأنها ممكن تفشل وترجع null بالتالى هكون
ضيعت ال pointer الأساسى بتاعى فلازم الاول اعمل
Check لو مش بـ null اعمله update

آخر حاجه لو عايزة اعمل dynamic allocation
على ال Stack هناك alloca()

هى زى ال malloc بس هى بتallocate فين ؟
من ال Stack تحديداً من ال frame بتاع ال stack
بتاع ال function اللى ندهت alloca
فبيروح يزود ال Stack ptr ويدين الحاجه الزيادة
اللى زودتها
ومش بيتعملها free

Lecture 4 - Part 2 / /

رب أشرح لى صدرى ويسرلى أمرى

ال environment list دى عبارة عن array of strings

بعرفها كدة name = Value (X=10 , X= Ahmed)

نبقى هى دى ال environment varibles

وكما نعمل fork ال Parent بيباصيهم لل Child أو هو بيورثهم يعنى ودى طريقة من طرق ال interprocess Communication

بس من مشاكلها انها One-Way ال Child ميعرفش يرجعهم لل Parent وكمان One time

لو ال Child غير فى ال environment vars بتاعته مفيش اى حد تانى هيشوفها غيره.

فى شوية library functions بتغير ال behaviour بتاعها لو فى environment var معين معمولة set أو هى بنفسها بتعمله set عشان نتصرف بشكل مختلف ودة بيخلينى اعرف اتحكم فى ال behaviour بتاعها من غير ما أغير فيه من جوا أو اعملها helink بعدين

نعمل export لل Var عشان ينتقل لأى Child عندى

وممكن أعمل unset عشان امسح اى واحد

ولو عايزة بس اخليها عند ال Child ومش عند ال Parent

$ Name = Value Program

وفي عندي Command اسمه env للي ممكن يعمل run
لـ program بيستخدم modified environment vars
انا اللي هديله ال option وال args اللي عايزه اعدلها
modify (read man page)

وفي printenv بتقولي مين ال environment vars
الموجودة عندي
واللي موجودين في Proc / PID / environ

ممكن اباصيهم لـ Program و استقبلهم من char** environ
ودة منتهي بـ null ودة هيكون global Varible

أو اني استقبلهم من تالت Var لل main

```
into main (int argc, char *argv[], char *envp)
```

local for main

Modifiging in C : Sent env un set env

وفي get env برضو (man page مش هكتب كل دة)
وفي put env دة هيحط Var لل environment Vars
تابعة لـ program والافضل set env